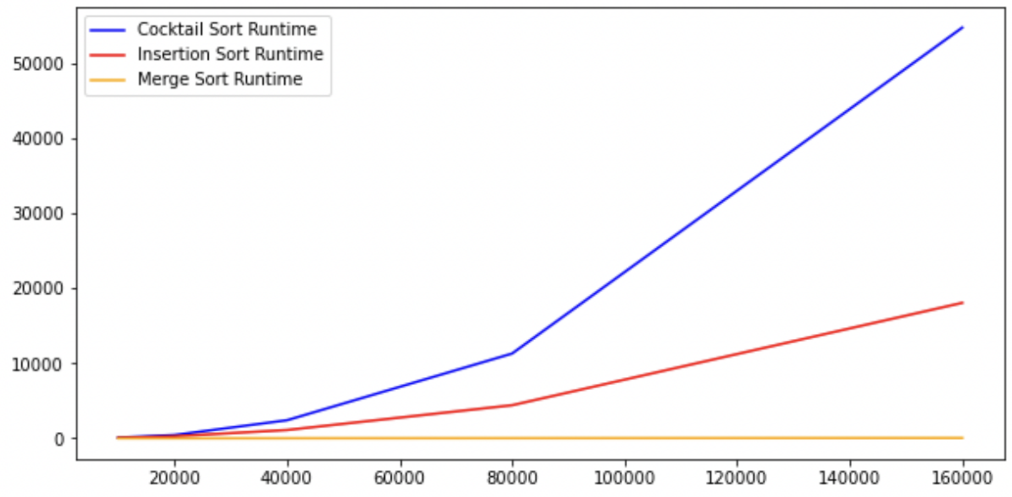
(1) Testing Cocktail Sort vs. Insertion Sort vs. Merge Sort

The time complexity of cocktail sort and insertion sort are O(n^2) while the slope of cocktail sort is larger than that of insertion sort, implying that the runtime of cocktail sort increases more when data size increases compared to insertion sort. While the complexity of merge sort is O(NlogN) which is smaller than O(n^2), the line of merge sort has much smaller slope than the two above ones. And since the y axis is based on the range of cocktail sort runtime, the line of merge sort is flattened so it looks like a horizontal line.

Data Size	Cocktail Sort	Insertion Sort	Merge Sort
10000	104	75	4
20000	443	279	5
40000	2414	1109	10
80000	11312	4416	22
160000	54792	18075	48

```
In [9]:  x = [10000, 20000, 40000, 80000, 160000]
         insertion = [75, 279, 1109, 4416, 18075]
         merge = [4, 5, 10, 22, 48]
         cocktail = [104, 443, 2414, 11312, 54792]
```

```
In [10]: plt.figure(figsize=(10, 5))
         plt.plot(x, cocktail, color="blue", label="Cocktail Sort Runtime")
         plt.plot(x, insertion, color="red", label="Insertion Sort Runtime")
         plt.plot(x, merge, color="orange", label="Merge Sort Runtime")
         plt.legend()
         plt.show()
```

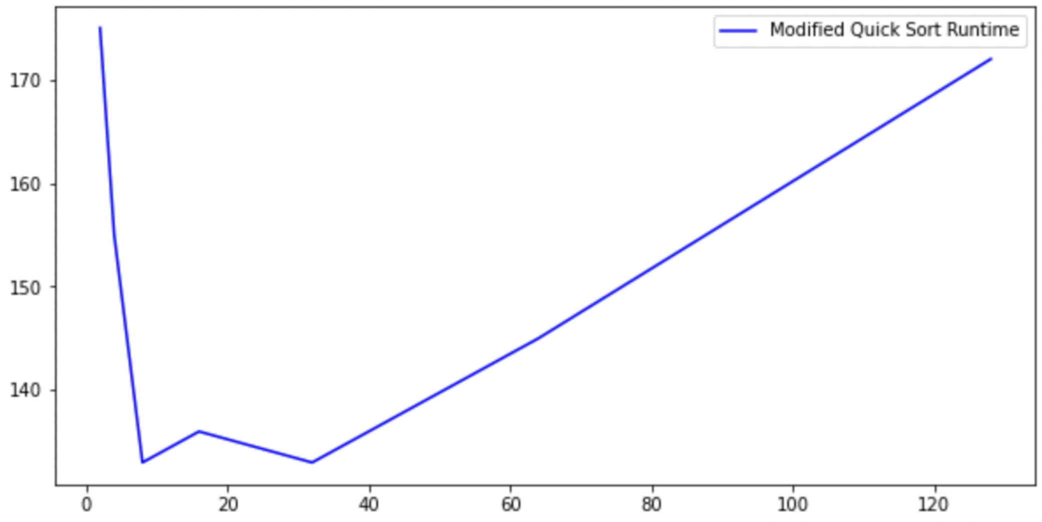


(2) Testing Modified QuickSort Cutoff Values

The graph is like the graph of x+1/x. It shows that the runtime first decreases dramatically when the cutoff is very small, then it starts to increase when the cutoff size increases. My best cutoff size is 8 and 32 since they have the same runtime.

Cutoff Size	Modified QuickSort
2	175
4	155
8	133
16	136
32	133
64	145
128	172

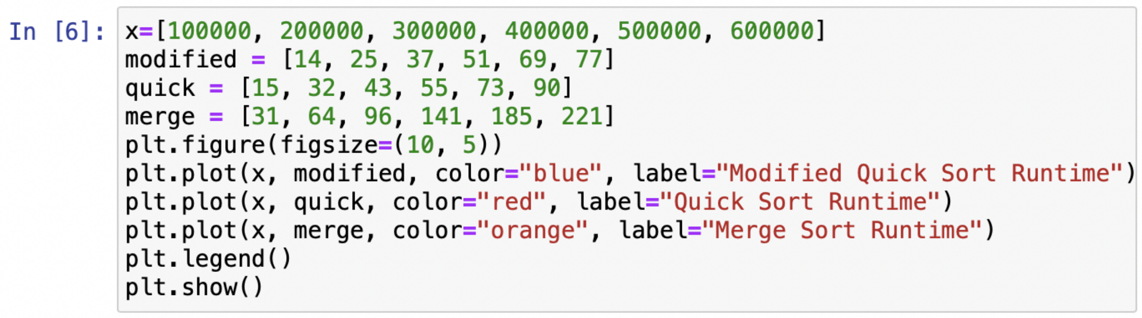
```
In [4]: x=[2, 4, 8, 16, 32, 64, 128]
        y=[175, 155, 133, 136, 133, 145, 172]
        plt.figure(figsize=(10, 5))
        plt.plot(x, y, color="blue", label="Modified Quick Sort Runtime")
        plt.legend()
        plt.show()
```



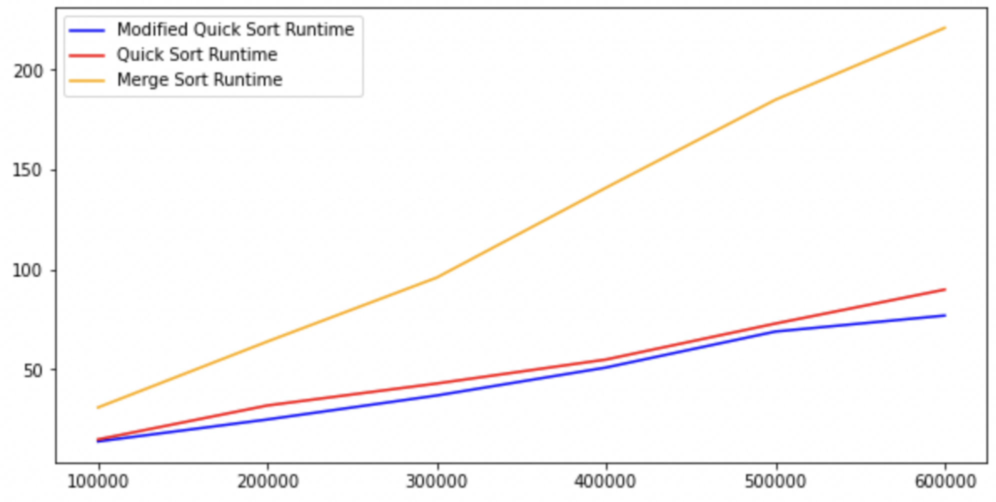
(3) Testing Traditional QuickSort vs. Modified QuickSort vs Merge Sort

The time complexity of merge sort, as stated in (1), is O(NlogN). While this time it's the one with the largest runtime, the slope is clear instead of a straight line in (1). The runtime of quicksort and modified quicksort is less than that of merge sort while being very close to each other. The time complexity of quicksort on average is O(NlogN) as well but we can see that its slope is smaller than that of merge sort, which shows that when data sizes increase, the increasing runtime for quicksort is less than that of merge sort. The runtime of modified quicksort is overall smaller than that of quicksort, showing that the modification is useful in decreasing runtime.

Data Size	QuickSort	Modified QuickSort	Merge Sort
100000	15	14	31
200000	32	25	64
300000	43	37	96
400000	55	51	141
500000	73	69	185
600000	90	77	221



```
In [6]: x=[100000, 200000, 300000, 400000, 500000, 600000]
        modified = [14, 25, 37, 51, 69, 77]
        quick = [15, 32, 43, 55, 73, 90]
        merge = [31, 64, 96, 141, 185, 221]
        plt.figure(figsize=(10, 5))
        plt.plot(x, modified, color="blue", label="Modified Quick Sort Runtime")
        plt.plot(x, quick, color="red", label="Quick Sort Runtime")
        plt.plot(x, merge, color="orange", label="Merge Sort Runtime")
        plt.legend()
        plt.show()
```

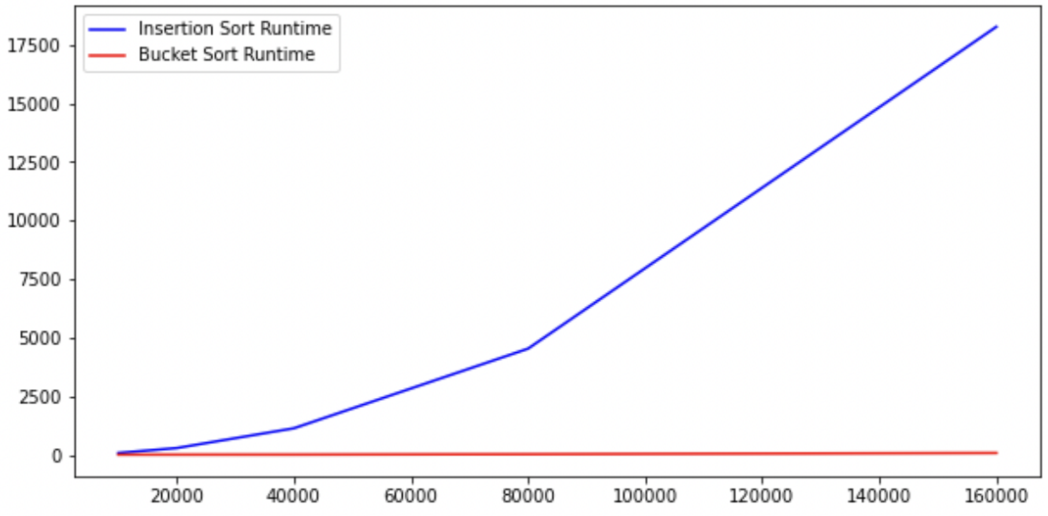


(4) Testing Insertion Sort vs. Bucket Sort

The time complexity of insertion sort is O(n^2) while that of bucket sort is O(n), so the slope of insertion sort is larger than that of bucket sort. It shows that when data size increases, insertion sort runtime increases more than bucket sort runtime.

Data Size	Insertion Sort	Bucket Sort
10000	77	5
20000	287	6
40000	1130	8
80000	4528	24
160000	18265	77

```
In [14]: x=[10000, 20000, 40000, 80000, 160000]
         insertion = [77, 287, 1130, 4528, 18265]
         bucket = [5, 6, 8, 24, 77]
         plt.figure(figsize=(10, 5))
         plt.plot(x, insertion, color="blue", label="Insertion Sort Runtime")
         plt.plot(x, bucket, color="red", label="Bucket Sort Runtime")
         plt.legend()
         plt.show()
```


Insertion Sort Runtime
Bucket Sort Runtime