Part 1: MergeSort

Questions

1. PSEUDOCODE:

```
1: BLOCK 1
2:     i++
3:     if i mod b = 0
4:         READ(X, i, b)
5: BLOCK 2
6:     j++
7:     if j mod b = 0
8:         READ(Y, j, b)
```

Algorithm 1: MergeSort BLOCKS

2. The merge function must call the `read` and `write` functions n/b times.

Part 2: Benchmarks

3. Bridges URL: https://bridges-cs.herokuapp.com/assignments/247106/joonsuh

4. Screenshots:

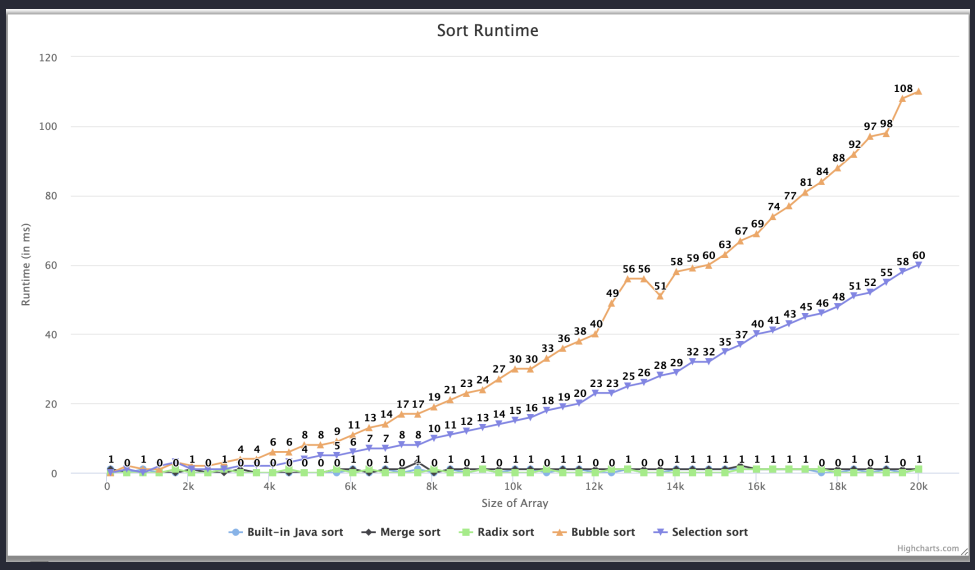

Figure 1: Linearly scaled input graph

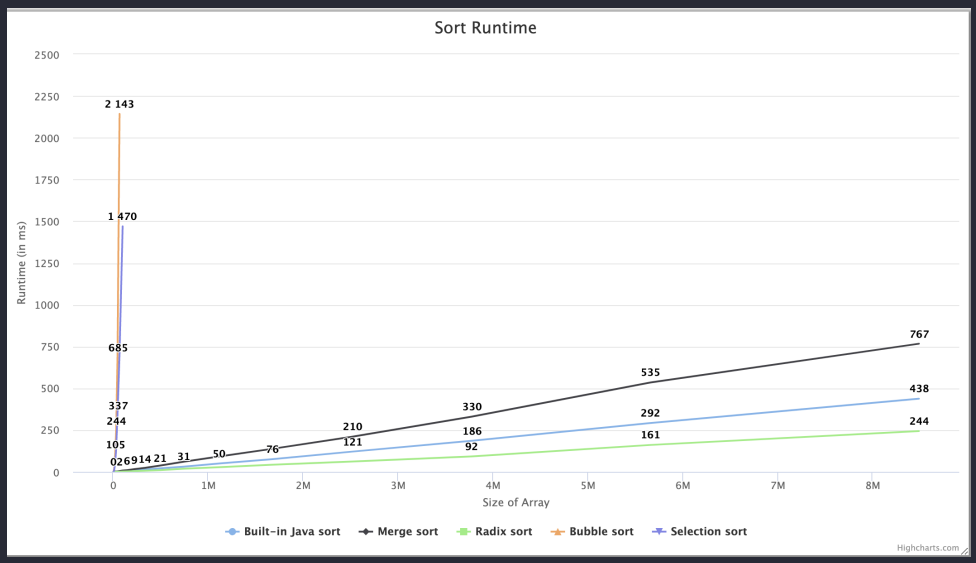

Figure 2: Geometrically scaled input graph

5. From slowest to fastest the performance of the algorithms (selection sort, bubblesort, mergesort, radix sort) the performance of the algorithms is

   selection sort = bubblesort $\rightarrow$ mergesort = built-in Java sort $\rightarrow$ radix sort

6. When looking at selection sort and bubblesort, the performance of the algorithms is

   bubblesort $\rightarrow$ selection sort

   where selection sort is slightly faster than bubblesort. Even though both algorithms have $O(n^2)$ time complexity, selection sort is faster than bubblesort because it performs fewer swaps in total i.e. it only swaps the minimum element with the first element of the array vs. swapping adjacent elements in bubblesort.