Info

Name: Eilidh McAlonan Matriculation number: 2560102

Running

- To see all tests being validated, run `src/TestSortingAlgorithms.java`'s main method

Extra

- Each `*Sort.java` file can be run individually to see their validation output if you like

QuickSort (3 way) explanation

- I modified `partition` to output a tuple (i, j) representing the range of indexes where the elements are equal to the pivot
- Instead of recurring to the left and right of a single index, the left slice is from *lower* to $i - 1$ and the right slice is from $j + 1$ to *upper*

Part 3 explanation

- I chose a heap sort (with a max heap) because it finds the maximum element in the heap each time the heap is rebuilt
  - The largest element is placed at the root of the heap
  - This means that I can take these elements `k` times and exit early
  - This avoids sorting the whole array before taking `k` elements

Running time analysis

- `buildHeap` is $O(n)$ as seen in lecture 8
- `heapify` is $O(\log n)$ as seen in lecture 8
- The loop in `topK` should only iterate `k` times. This loop calls `heapify`, so its complexity is $O(k \log n)$
- The heap has to be built at the start, so the overall complexity is $O(k \log n + n) \equiv O(k \log n)$

Results

Insertion sort

Elements	Run time (ms)
int1000	0
int20k	17
int500k	6911
dutch	6699
intBig	27494

Merge sort

Elements	Run time (ms)
int1000	0
int20k	2
int500k	56
dutch	32
intBig	77

QuickSort (right pivot)

Elements	Run time (ms)
int1000	1
int20k	Stack overflow
int500k	Stack overflow
dutch	306
intBig	Stack overflow

QuickSort (hybrid)

k = 8

Elements	Run time (ms)
int1000	1
int20k	Stack overflow
int500k	Stack overflow
dutch	372
intBig	Stack overflow

k = 16

Elements	Run time (ms)
int1000	2
int20k	Stack overflow
int500k	Stack overflow
dutch	392
intBig	Stack overflow

QuickSort (median)

I failed to get this one working

QuickSort (3 way)

Elements	Run time (ms)
int1000	2
int20k	Stack overflow
int500k	Stack overflow
dutch	103
intBig	Stack overflow

Discussion

- The 3-way QuickSort was the fastest of the QuickSorts
- Increasing `k` in the hybrid QuickSort did not yield better performance
- Merge Sort consistently outperforms Insertion Sort, albeit at the cost of memory footprint
- Despite being the same size as `int500k`, all algorithms performed better on `dutch`
  - This reveals that the stack overflow errors may be related to hitting the worst case
- Merge Sort outperformed all QuickSorts. This may be due to my implementation recurring unnecessarily (as shown by the stack overflow errors)

Stack Overflows.

- These errors occur inconsistently - they don't appear with small inputs or `dutch.txt`
- I was unable to reproduce this issue with random data
  - `QuickSortRight` runs with 10 million elements
- This issue is in all variants, despite the fact that `QuickSort3Way` only shares `Sorter.swap` with the other versions
- I tried merging `pickPivot` into `partition` to reduce the number of method calls per `sortInner` call