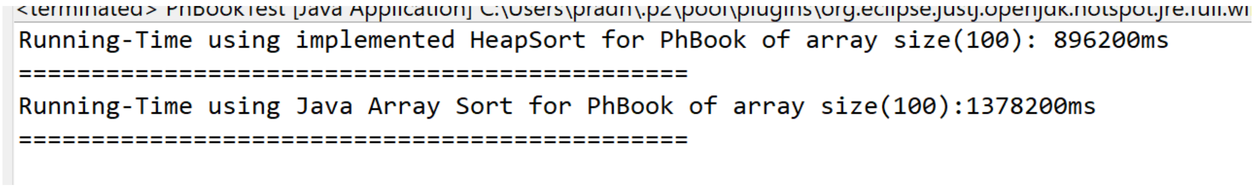
Big-O Analysis of my HeapSort Algorithm in Project 6:

The method heapSort *[public void heapSort(Person[] arr, int startIndex, int endIndex)]* has a worst, best, and average time complexity of O(nlogn). Where n represents the number of person nodes in the heap.

- My heapSort method uses the helper method [buildHeap] which has a worst case time complexity of O(n) to turn the array into a heap, where n is the number of person nodes in the heap.
- It takes a worst case time complexity of O(log n) to remove the largest remaining item
- And has n above removals
- This equals the overall time worst, best, and average case time complexity of O(nlogn) of my heapSort algorithm

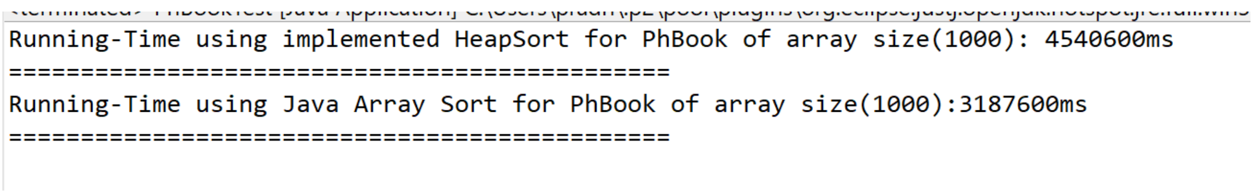
The timestamps for executions for the arrays of size 100, 1000, 10,000, 100,000, and 1,000,000 by running the PhBookTest.java file

For Array Size 100 :



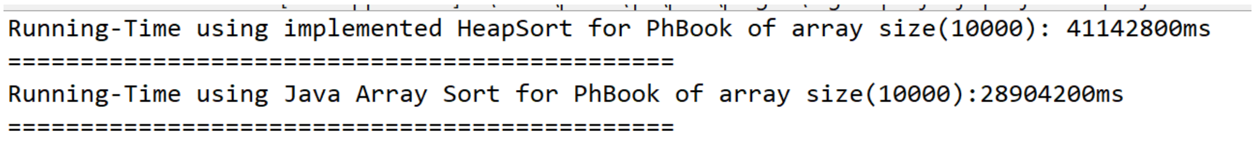
```
<terminated> PhBookTest [Java Application] C:\Users\pradn\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.wi
Running-Time using implemented HeapSort for PhBook of array size(100): 896200ms
===============================================
Running-Time using Java Array Sort for PhBook of array size(100):1378200ms
===============================================
```

For Array Size 1000:



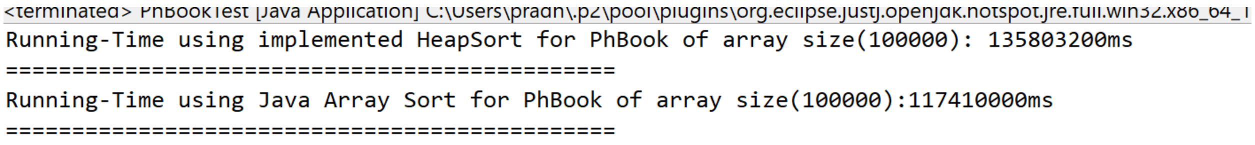
```
Running-Time using implemented HeapSort for PhBook of array size(1000): 4540600ms
===============================================
Running-Time using Java Array Sort for PhBook of array size(1000):3187600ms
===============================================
```

For Array Size 10000:



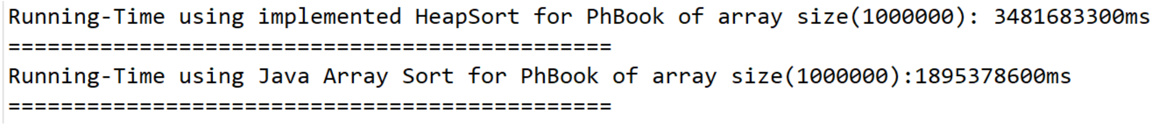
```
Running-Time using implemented HeapSort for PhBook of array size(10000): 41142800ms
===============================================
Running-Time using Java Array Sort for PhBook of array size(10000):28904200ms
===============================================
```

For Array Size 100000:



```
<terminated> PhBookTest [Java Application] C:\Users\pradn\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_1
Running-Time using implemented HeapSort for PhBook of array size(100000): 135803200ms
==============================================
Running-Time using Java Array Sort for PhBook of array size(100000):117410000ms
==============================================
```

For Array Size 1000000:



```
Running-Time using implemented HeapSort for PhBook of array size(1000000): 3481683300ms
==============================================
Running-Time using Java Array Sort for PhBook of array size(1000000):1895378600ms
==============================================
```

Conclusions:

It is observed that my implementation of heapsort runs faster with a smaller array size of 100, however for larger arrays 1000 through 1000000, the sort() method from the java.utils.Arrays class is faster.