CSC 242
Lab 2
Modify the `quicksort` function (found in the file ***testquicksort.py***), so that it calls `insertionSort` (found in the file ***algorithms.py***) to sort any sublist whose size is less than 50 (threshold) items. Compare the performance of this version with that of the original one, using data sets of 50, 500, 5000 items. Then, adjust the threshold for using the insertion sort to determine an optimal setting.
Is performance of this modified quicksort dependent upon the threshold?
Does the size of the list affect the modification?
When should the insertion sort be used?
Are the number of comparisons and exchanges affected by the modification?

Here is an example driver:

```
import random
def main(size = 50, sort = quicksort):
    lyst = []
    p = Profiler()
    for count in range(size):
        lyst.append(random.randint(1, size + 1))
    print(lyst)
    p.test(sort, lyst, size, unique = True, comp = True, exch = True, trace = False, mod = False)

if __name__ == "__main__":
     main()
```

and it’s output:

```
[18, 9, 31, 23, 8, 34, 43, 34, 33, 33, 17, 19, 51, 14, 3, 16, 31, 21, 48, 44, 29, 24, 13, 26, 31, 18, 37,
11, 48, 27, 26, 38, 7, 40, 24, 45, 29, 11, 46, 48, 8, 21, 15, 43, 7, 42, 47, 17, 44, 51]
[3, 7, 7, 8, 8, 9, 11, 11, 13, 14, 15, 16, 17, 17, 18, 18, 19, 21, 21, 23, 24, 24, 26, 26, 27, 29, 29, 31,
31, 31, 33, 33, 34, 34, 37, 38, 40, 42, 43, 43, 44, 44, 45, 46, 47, 48, 48, 48, 51, 51]
Problem size: 50
Elapsed time: 0.001
Comparisons:  251
Exchanges:    213
Modified with insertion sort: False
```