CS2302 Data Structures

Fall 2019

Lab Report #2

Due: March 10th, 2019

Professor: Olac Fuentes

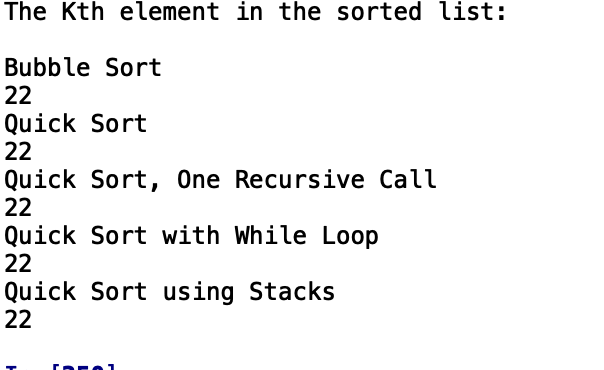
TA: Anindita Nath

**Introduction**

The problem in this lab is to find the Kth element amongst lists that are sorted. We have to sort the list L using bubble sort, quicksort, modified quicksort using only one recursive call, a quick sort using a while loop and no recursion, and a quicksort using stacks.

**Proposed Solution Design and Implementation**

Design -



This is the output that the user will see when running the program. I added a method that resets the list each time the individual methods so that it doesn’t send an already sorted list.

Operation #1: I implemented the partition method that would return the index of the pivot which I then use to target elements below and above the pivot. All of the quicksort methods use the partition method, each method works around this method to sort the list. Each sorting method has its own method that is called with all of the same attributes that outputs the Kth index of the list of the sorted list or a portion of the sorted list.

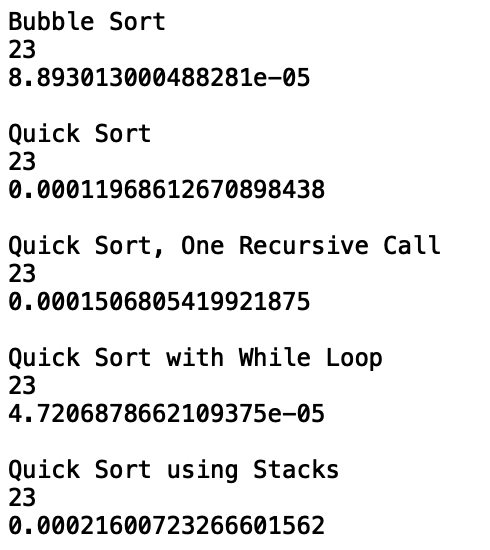
Operation #2: The second part of the lab asked for a quick sort which uses stacks which still uses the same attributes as the other quicksort methods, the method still outputs the Kth element of the list. The while loop quick sort doesn’t use any recursion and uses the partition method heavily to sort and output the Kth element of the list that’s passed to it.

**Experimental Results**

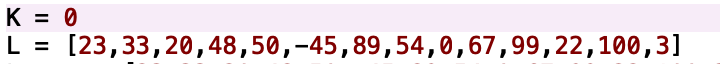
The bubble sort method has a running time complexity of O(n^2), the quicksort using recursion results in recurrence relation T(n) = 2T(n/2) + n resulting in a time complexity of O(nlogn), the quicksort that only uses one recursive call has a recurrence relation of T(n) = T(n/2) + n resulting in time complexity of O(n). The quick sort using stacks has a time complexity of O(n) and the quick sort using while loop has the time complexity O(n).

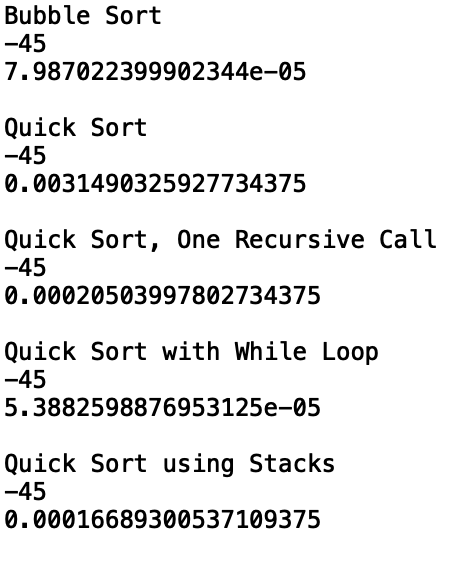
Case 1:





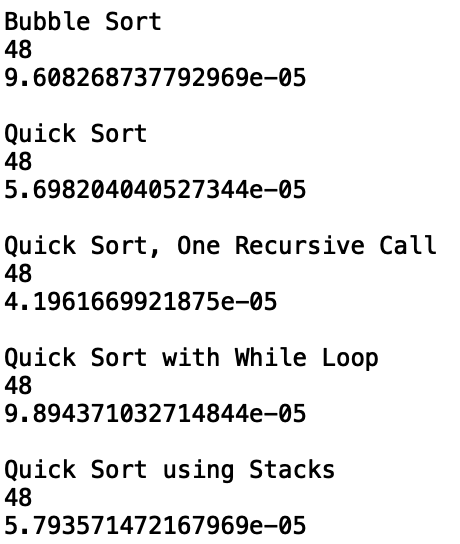
Case 2:





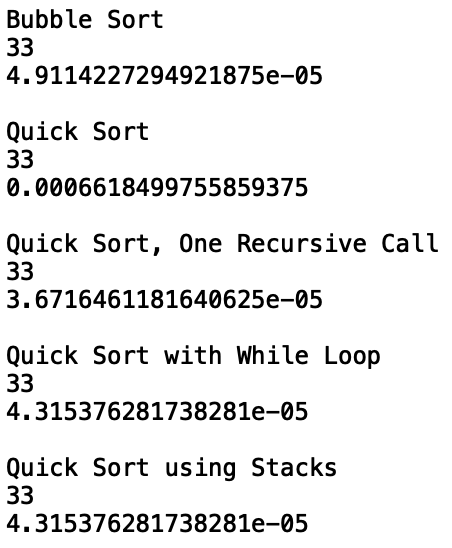
Case 3:





Case 4:

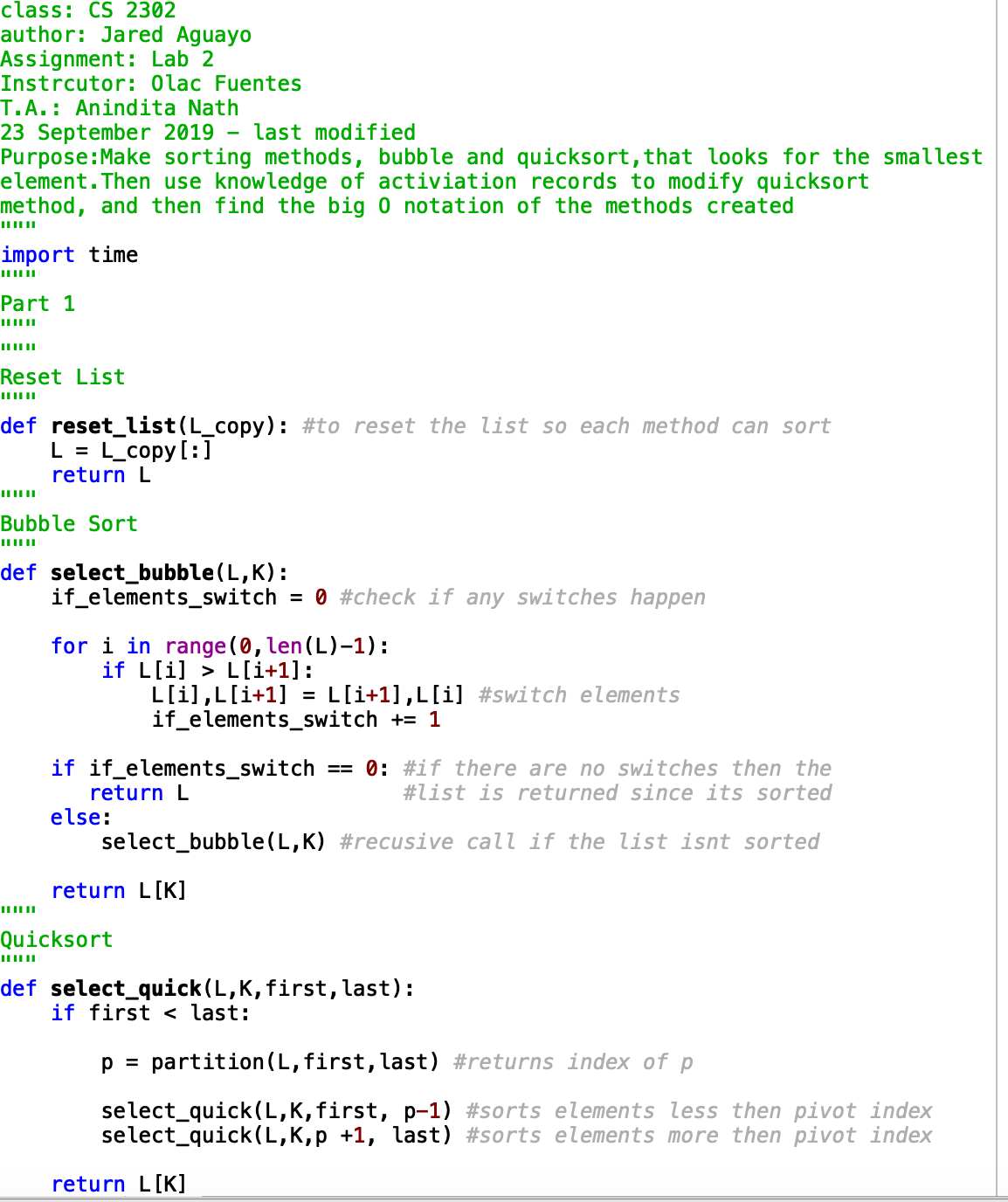
****

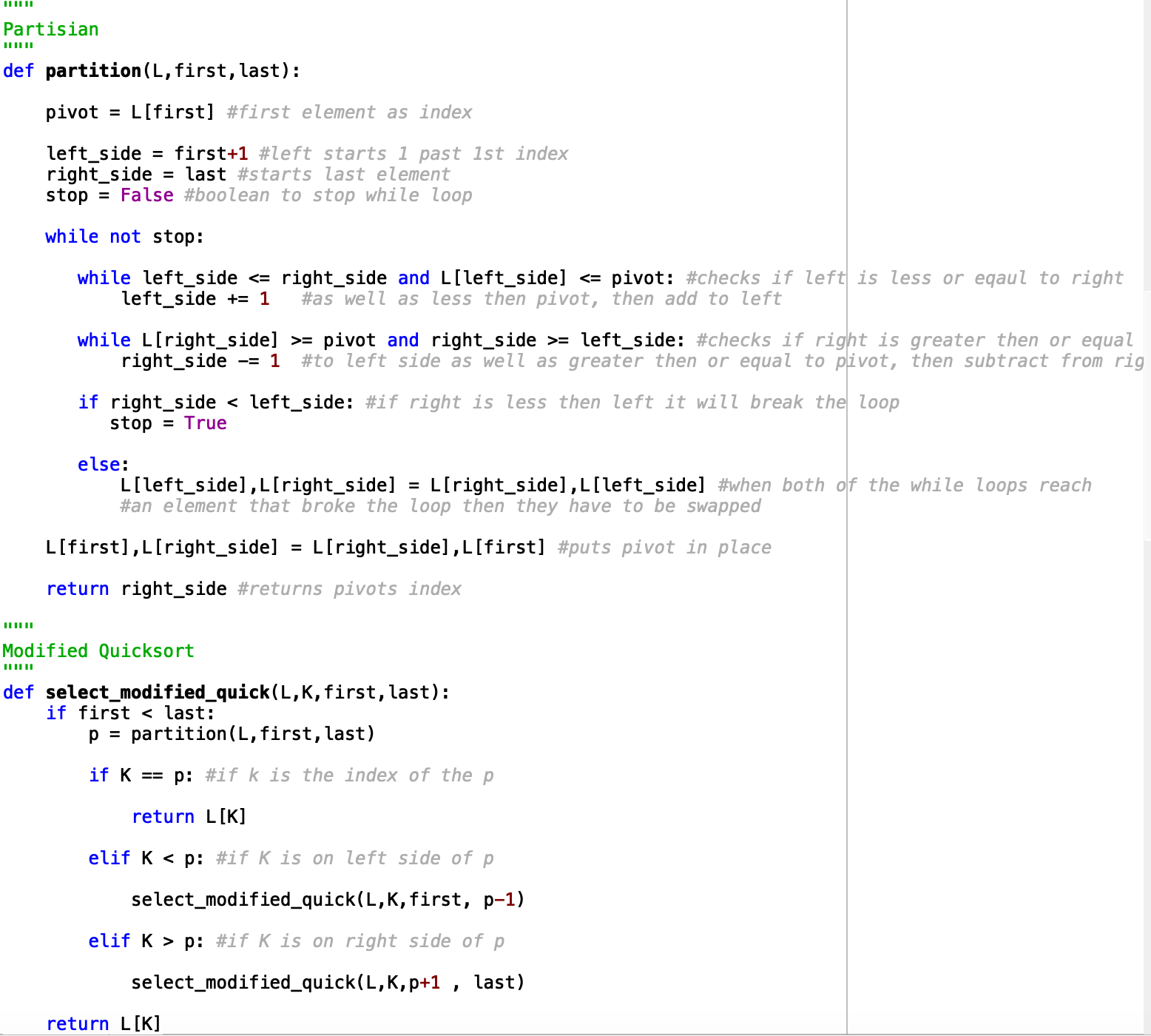
****

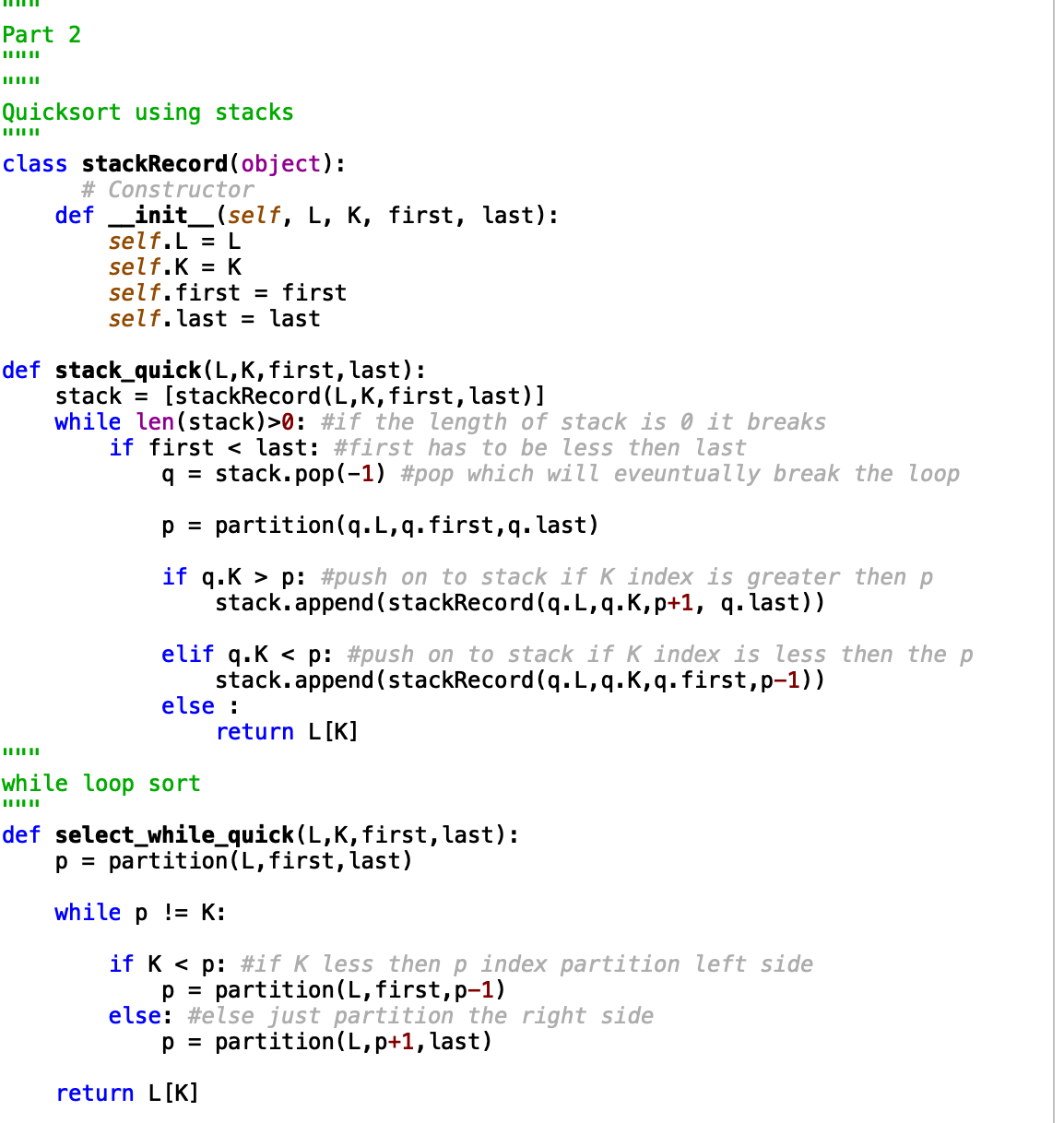
**Conclusions**

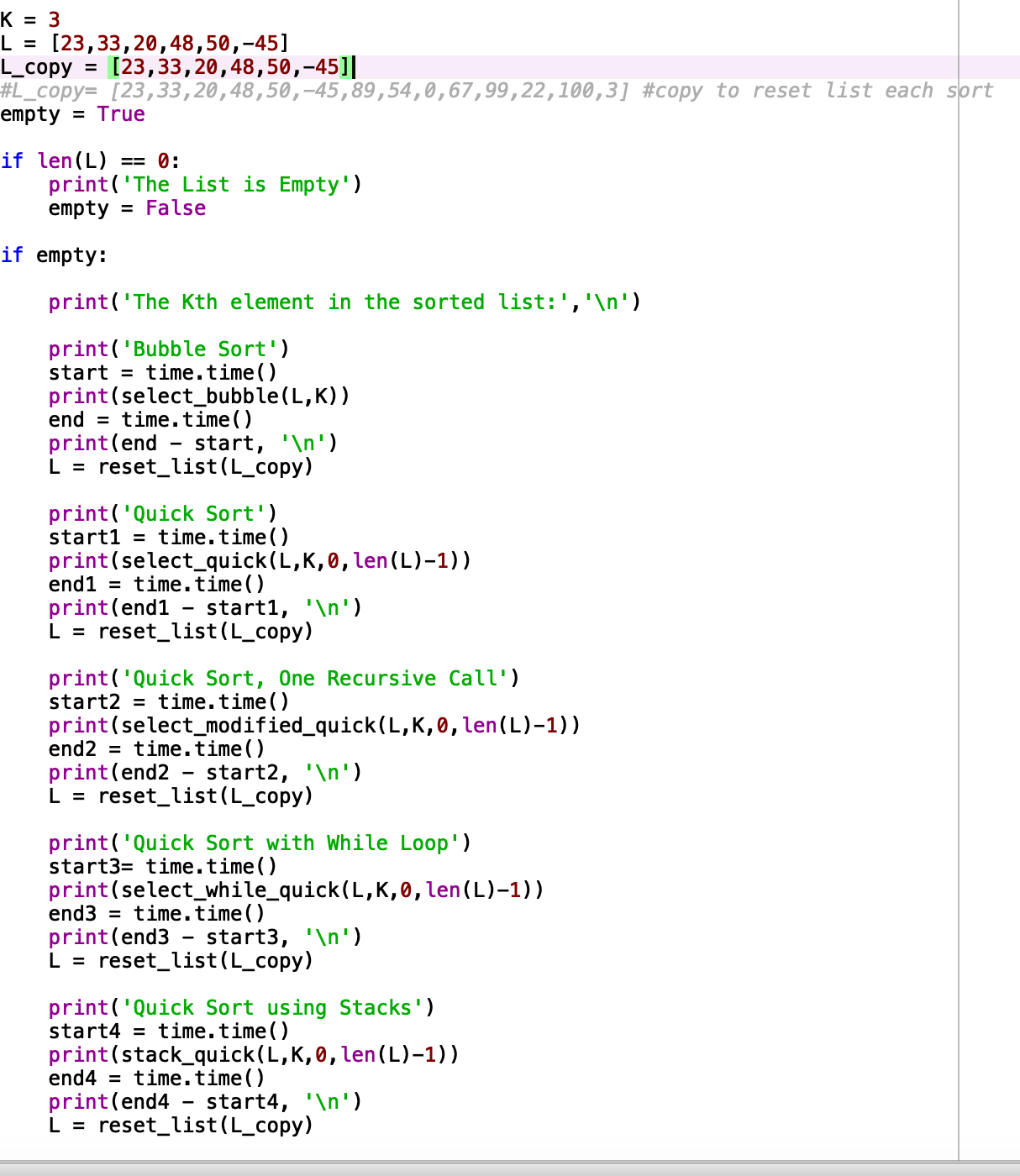
I learned about different ways to implement sorting methods that affects the time complexity when thinking about what the most efficient way is to implement a quick sort method.

**Appendix**









I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class