**Pseudocode**

**START** program

**SET** integer *choice* = 0

**CREATE** a LinkedList object named *bidList*

**CREATE** vector *bid* to BID objects

**CREATE** time\_t *ticks*

**WHILE** *choice* not equal to 9

**OUTPUT** menu options

**INPUT** *choice*

**IF***choice* equals 1

**SET** *bid* equal to the getBid() function

**INVOKE** Append function on *bidList* with *bid* as the argument

**INVOKE** displayBid() function, giving *bid* as the argument

**ELSE IF** choice equals 2

**SET** *ticks* equal to the current system time

**INVOKE** loadBids() function, sending csvPath, and a pointer to *bidList* as

arguments

**OUTPUT** *bidList* size + “ bids read”

**SET** *ticks* equal to current time - *ticks*

**OUTPUT** “time: “ + *ticks* + “ milliseconds”

**OUTPUT** “time: “ + *ticks* \* 1.0 / CLOCKS\_PER\_SEC + “ seconds”

**ELSE IF** choice equals 3

**INVOKE** printList() function on *bidList*

**ELSE IF** *choice* equals 4

**SET** *ticks to* current system time

**SET** *bid* equal to the Search() function invoked on *bidList* with *bidkey* as the argument

**SET** *ticks* to current system time - current ticks value

**IF** *bid*’s bidId is not empty

**INVOKE** displayBid() function sending *bid* as the argument

**ELSE**

**OUTPUT** “Bid Id” + *bidkey* + “ not found.”

**OUTPUT** “time: “ + *ticks* + “ clock ticks”

**OUTPUT** “time: “ + *ticks* \* 1.0 / CLOCKS\_PER\_SEC + “seconds”

**ELSE IF** *choice* equals 5

**INVOKE** Remove() function on *bidList* sending *bidkey* as the argument

**ELSE IF** *choice* equals 9

**EXIT** loop

**ELSE**

**CONTINUE** loop

**OUTPUT** “Good bye”

**END** program

**Reflection**

There are many different types of data structures. One such structure is a singly linked list. The singly linked list that the above psuedocode refers too, is being used to structure a list of eBids. The linked list has multiple functions that allow many possibilities. The functions include appending new bids, loading the bid file into the structure, a print function to show all the bids, a search function to find any particular bid, and a remove function that allows for removal while keeping the connection of all bids intact.

Creating a linked list can be a challenge. While coding there is a lot of connectivity that must stay intact. The logic is easy to follow on paper, but when it’s time to code little things can be missed. Whenever a node is removed the node before it’s next pointer must be moved to the node after the one being removed to continue the connectivity.

Vahid, F. (2019). CS300: Data Structure and Algorithms. ZyBooks. Retrieved July 2, 2023 from <https://learn.zybooks.com/zybook/CS-300-X6110-OL-TRAD-UG.23EW6>