# CS 300 Linked List and Pseudocode

Robert Umland

Robert.Umland@SNHU.EDU

Southern New Hampshire University

**Code Reflection**

This week’s project was to take a .csv file of eBid auction sales, create a linked list from the contents and create a new node in the list. This will then be found via a search and deleted from the linked list. Both appending the list and prepending the list are options in the code, though the main() function only call the append. Coding this week was much easier than last week, the cues in the Fixme comments were immensely useful. I spent time reworking the code to try the prepend as well as the append to ensure my work was correct. I even added more nodes to test the search was working. My biggest challenge this week was ensuring that all the code was working to expectations and the commenting was sufficient.

**Pseudocode**

// Define a structure to hold bid information

structure Bid

string bidId

string title

string fund

double amount

constructor Bid()

amount = 0.0

// Define Linked-List class

class LinkedList:

// Define internal structure for list entries, housekeeping variables

struct Node:

Bid bid

Node \*next

Node() constructor initializes next to nullptr

Node(Bid aBid) constructor initializes bid and sets next to nullptr

Node\* head

Node\* tail

int size initialized to 0

LinkedList() constructor sets head and tail to nullptr

LinkedList() destructor deletes all nodes in the list

void Append(Bid bid):

Create new node

If head is nullptr:

Set head and tail to new node

Else:

Set tail's next to new node

Set tail to new node

Increment size

void Prepend(Bid bid):

Create new node

If head is nullptr:

Set head and tail to new node

Else:

Set new node's next to head

Set head to new node

Increment size

void PrintList():

Set current to head

While current is not nullptr:

Print current bid details

Set current to current's next

void Remove(string bidId):

If head is nullptr, return

If head's bidId matches bidId:

Set temp to head

Set head to head's next

Delete temp

Decrement size

Return

Set current to head

While current's next is not nullptr and next's bidId does not match bidId:

Set current to current's next

If current's next is not nullptr:

Set temp to current's next

Set current's next to next's next

If temp is tail:

Set tail to current

Delete temp

Decrement size

Bid Search(string bidId):

Set current to head

While current is not nullptr:

If current's bidId matches bidId, return current bid

Else set current to current's next

Create empty bid and return

int Size():

Return size

\\ Define static methods for testing

void displayBid(Bid bid):

Print bid details

Bid getBid():

Create bid

Prompt user for bid details and set bid properties

Return bid

void loadBids(string csvPath, LinkedList \*list):

Initialize CSV Parser

Try reading rows of CSV file

For each row, initialize bid and set properties from CSV data

Append bid to list

double strToDouble(string str, char ch):

Remove character from string and convert to double

int main(int argc, char\* argv[]):

Process command line arguments

Initialize variables

Create LinkedList

Create Bid

While choice is not 9:

Display menu

Get user choice

Switch on choice:

case 1:

Get bid from user and append to list

Display bid

case 2:

Start timer

Load bids from CSV file

Print number of bids read

Print elapsed time in ticks and seconds

case 3:

Print list

case 4:

Start timer

Search for bid in list

Print bid details or not found message

Print elapsed time in ticks and seconds

case 5:

Remove bid from list

Print goodbye message

Return 0