# CS 300 Vector Sorting Reflection and Pseudocode

Robert Umland

Robert.Umland@SNHU.EDU

Southern New Hampshire University

**Code Reflection**

This week’s project was designed to take a .csv file of eBay auction sales and sort it by title. It uses We had to design the two sorting methods and add them to the menu options in main(). The two sorting methods were Quicksort and Selection Sort. I used what I learned this week and from some previous classwork to create the two sorting methods. Probably the biggest challenge was I haven’t coded C++ in a few months. Since any language atrophies when not used (spoken or written), I was rusty. It took some time to get reacquainted with it, and I’m still working on getting it straight.

**Pseudocode**

// Define a structure to hold bid information

structure Bid

string bidId

string title

string fund

double amount

constructor Bid()

amount = 0.0

// Display the bid information

function displayBid(bid)

print bid.bidId, bid.title, bid.amount, bid.fund

// Prompt user for bid information

function getBid()

Bid bid

print "Enter Id: "

read bid.bidId

print "Enter title: "

read bid.title

print "Enter fund: "

read bid.fund

print "Enter amount: "

string strAmount

read strAmount

bid.amount = strToDouble(strAmount, '$')

return bid

// Load bids from a CSV file

function loadBids(csvPath)

print "Loading CSV file ", csvPath

list of Bid bids

CSV file = CSV(csvPath)

try

for i from 0 to file.rowCount - 1

Bid bid

bid.bidId = file[i][1]

bid.title = file[i][0]

bid.fund = file[i][8]

bid.amount = strToDouble(file[i][4], '$')

append bid to bids

catch Error e

print e.message

return bids

// Partition the bids for quicksort

function partition(bids, begin, end)

int low = begin

int high = end

int middlePoint = low + (high - low) / 2

string pivot = bids[middlePoint].title

while true

while bids[low].title < pivot

low++

while pivot < bids[high].title

high--

if low >= high

return high

swap bids[low] with bids[high]

low++

high--

// Perform quicksort on bids

function quickSort(bids, begin, end)

if begin < end

int pivotIndex = partition(bids, begin, end)

quickSort(bids, begin, pivotIndex - 1)

quickSort(bids, pivotIndex + 1, end)

// Perform selection sort on bids

function selectionSort(bids)

for i from 0 to bids.size - 1

int minIndex = i

for j from i + 1 to bids.size - 1

if bids[j].title < bids[minIndex].title

minIndex = j

if minIndex != i

swap bids[i] with bids[minIndex]

// Convert a string to double

function strToDouble(str, ch)

remove all instances of ch from str

return atof(str)

// Main function

function main(argc, argv)

string csvPath

if argc == 2

csvPath = argv[1]

else

csvPath = "CS 300 eBid\_Monthly\_Sales.csv"

print "CSV Path: ", csvPath

list of Bid bids

clock ticks

int choice = 0

while choice != 9

print menu options

read choice

switch choice

case 1:

ticks = clock()

bids = loadBids(csvPath)

print bids.size, " bids read"

ticks = clock() - ticks

print "time: ", ticks, " clock ticks"

print "time: ", ticks / CLOCKS\_PER\_SEC, " seconds"

case 2:

for i from 0 to bids.size - 1

displayBid(bids[i])

case 3:

ticks = clock()

selectionSort(bids)

print bids.size, " bids sorted using selection sort"

ticks = clock() - ticks

print "time: ", ticks, " clock ticks"

print "time: ", ticks / CLOCKS\_PER\_SEC, " seconds"

case 4:

ticks = clock()

quickSort(bids, 0, bids.size - 1)

print bids.size, " bids sorted using quicksort"

ticks = clock() - ticks

print "time: ", ticks, " clock ticks"

print "time: ", ticks / CLOCKS\_PER\_SEC, " seconds"

print "Good bye."