Vector Sorting Assignment

# Code Reflection

The purpose of this program is to load bid data from CSV files into a vector, then allow the user to sort the bids by title using two algorithms: Selection Sort and Quick Sort. The implementation highlights the performance differences between O(n^2) sorting (Selection Sort) and O(n log n) sorting (Quick Sort).   
  
Selection Sort is straightforward but inefficient for large data sets. It iterates through the list, repeatedly finding the minimum element and swapping it. This method works well for small datasets but is very slow when the number of records grows large.   
  
Quick Sort, on the other hand, uses a divide-and-conquer approach. By partitioning the data around a pivot and recursively sorting the partitions, it achieves significantly better performance on large data sets. The results matched expectations, with Quick Sort sorting 12,000+ bids in a fraction of a second compared to over 10 seconds for Selection Sort.   
  
The project demonstrates the importance of choosing the correct algorithm and data structure for performance. It also reinforced good programming practices such as modular functions, consistent naming, and clear output for timing results.

# Pseudocode

STRUCT Bid: title, fund, amount  
  
FUNCTION loadBids(path):  
 for each row in CSV:  
 build Bid from ArticleTitle, Fund, WinningBid  
 append to bids  
 return bids  
  
FUNCTION selectionSort(bids):  
 for i = 0 to n-2:  
 min = i  
 for j = i+1 to n-1:  
 if bids[j].title < bids[min].title:  
 min = j  
 swap bids[i], bids[min]  
  
FUNCTION partition(bids, begin, end):  
 pivot = bids[end].title  
 i = begin - 1  
 for j = begin to end-1:  
 if bids[j].title <= pivot:  
 i++  
 swap bids[i], bids[j]  
 swap bids[i+1], bids[end]  
 return i+1  
  
FUNCTION quickSort(bids, begin, end):  
 if begin >= end: return  
 p = partition(bids, begin, end)  
 quickSort(bids, begin, p-1)  
 quickSort(bids, p+1, end)  
  
MAIN:  
 while choice != 9:  
 display menu  
 case 1: loadBids, report time  
 case 2: display all bids  
 case 3: selectionSort, report time  
 case 4: quickSort, report time  
 case 9: exit

# Flowchart

The following flowchart shows the program logic:

