CS 161B: Programming and Problem Solving II

Assignment 3: Array Statistics

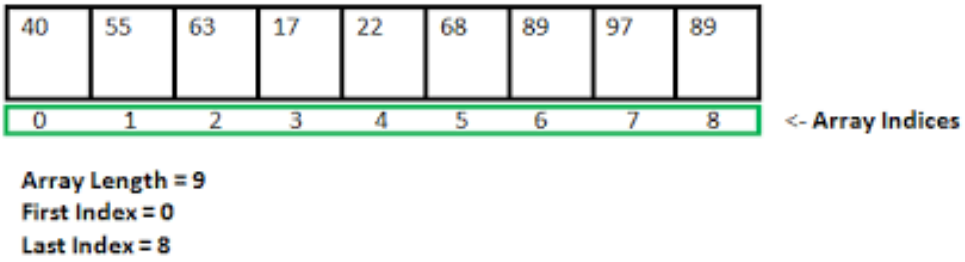
Academic Integrity

You may NOT, under any circumstances, begin a programming assignment by looking for completed code on StackOverflow or Chegg or any such website, which you can claim as your own. Please check out the Student Code of Conduct at PCC.

The only way to learn to code is to do it yourself. The assignments will be built from examples during the lectures, so ask for clarification during class if something seems confusing. If you start with code from another source and just change the variable names or other content to make it look original, you will receive a zero on the assignment.

I may ask you to explain your assignment verbally. If you cannot satisfactorily explain what your code does, and answer questions about why you wrote it in a particular way, then you should also expect a zero.

Arrays store data contiguously by representing data with a common name and distinguishing it by its index. This is like a parking garage: the garage stores vehicles; all parking spaces have a common name (the name of the garage), and a specific parking space number identifies each parked vehicle. This is also true of arrays. Think of arrays as parking garages for our data!



Purpose

In this assignment, you will be writing a program that stores a list of positive integers from input into an array and calculates the minimum, maximum, mean, median, mode, and whether the array is a palindrome. A negative integer indicates the end of the input and is not stored in the array.

After completing this assignment you will be able to:

- Define an array to hold multiple pieces of information of the same data type
- Use an appropriate loop to read input from the user into the array
- Use a for loop to manipulate the array without going out of bounds
- Pass an array and the size to a function and manipulate the array
- Do input data validation and use the correct exit condition to exit out of the loop
- Convert an algorithm to code

Task

Write a program to calculate the minimum, maximum, mean, median, mode, and whether an array is a palindrome. Write the following functions and call them in the same order as instructed in the `main()` function to accomplish this Task.

- ❏ `void readInput(int list[], int &count);` Reads a list of positive integers from the user. A negative integer indicates the end of the input and is not stored in the array. (Use an appropriate loop!! Check out this example code - a for loop is count controlled, and a while loop is condition based.)
- ❏ `void maxmin(int list[], int count, int &max, int &min);` Use a loop to process each array element and return the minimum and maximum values to main by reference. These values should be printed in `main()`.
- ❏ `double avgArray(int list[], int count);` Use a loop to sum all the array elements and calculate the mean (or average). Return the mean to `main()` and in `main()` print the average with one decimal place using `cout << fixed << setprecision(1);`.
- ❏ `bool isPalindrome(int list[], int count);` Use a loop to determine if the array is a palindrome, meaning values are the same from front to back and back to front. Output "true" or "false".
- ❏ `void sort (int list[], int count);` Sort the array using the given sorting algorithm. This is called Selection Sort. Use only this algorithm to sort your list. To see how the Selection Card Sort Algorithm works, watch this video from Virginia Tech.
- ❏ Watch this Python Video to help you with the sorting algorithm. Try this Selection Sort Animation by Y. Daniel Liang.

```
procedure selection sort
   list  : array of items
   count : size of list
   for i = 0 to count - 1
   /* set current element as minimum*/
     min = i
     /*go through the list and find the smallest
element*/
```

```
            for j = i+1 to count
                  if list[j] < list[min] then
                        min = j;
                  end if
            end for
            /* swap the minimum element with the current
      element
            If they are not the same element*/
            if indexMin != i  then
                  swap list[min] and list[i]
            end if
         end for
      end procedure
```

- `double median(int list[], int count);` After sorting, write this function to identify the median. The median is located in the middle of the array if the array's size is odd. Otherwise, the median is the average of the middle two values. Return the median to `main()` and output in `main()` with one decimal place.
- **Extra!! Extra!!! No grade associated with this extra - but by doing this extra function you will know that you have truly set your bar high to practice more with arrays!!**
  - **Challenge!** Identify the mode after the array is sorted in ascending order. The mode is the value that appears most frequently. Assume only one mode exists. ***Hint:*** Use a loop to process each array element, looking for the longest sequence of identical values. Remember the list is sorted - that means it is slightly easier to find the mode.
- **Assume the array will always contain fewer than 20 integers. You must not let the user enter more than 20 integers.**
- Open the Algorithmic Design Document, make a copy, and follow the steps to create your algorithm.
- You must express your algorithm as **pseudocode.**
- Your program must have function prototypes. Place the prototypes for your functions globally, after your #includes. **All functions must be implemented after main() and follow above mentioned requirements**.
- You may not use a while true loop and break statements in the loop.
- Your program must check for valid input data. Use loops to repeat until you get valid data from the user. Check out the sampleA01.cpp that has a **readInt function** that can be used to check for valid data. You can call this function to read integers for the array.
- Try not to have any redundant code (repeated code) in your program. That is the purpose of functions.
- Use constants where appropriate, for example - the capacity of the array.

- ❏ **Do not use vectors for this program. Use only the concepts and functions we have learned so far.**

Criteria for Success

- ❏ Use the below sample run as a guide to test your program.

```
Welcome to my Array Statistics program!
Please enter the values for the array (negative number to end
input:): 2 2 5 6 7 7 -3

Your stats are as below:
Minimum: 2
Maximum: 7
Mean: 4.8
Palindrome: false
Median: 5.5

Thank you for using my Array Statistics program!!
```

```
Welcome to my Array Statistics program!
Please enter the values for the array (negative number to end
input:): 1 2 3 4 5 4 3 2 1 -6

Your stats are as below:
Minimum: 1
Maximum: 5
Mean: 2.8
Palindrome: true
Median: 3

Thank you for using my Array Statistics program!!
```

- ❏ Complete zyBooks sections **11.1 to 11.11 CS161B: Arrays**, and **zyLabs 11.15 and 11.16**
- ❏ Complete all sections of your Algorithmic Design Document.
- ❏ You may use any IDE to write your code. If your Transfer University is PSU, check out the Special Note For PSU Transfer section in the Syllabus.
- ❏ Please bookmark the PCC Linux and Vim Manual - this will become a frequently used reference!
- ❏ Include **pseudocode** in part d of the design document.
- ❏ **Follow these Coding Construct Requirements:**
  - ❏ Do not use any vectors or containers for this program. Use only the concepts we have learned so far. Use only the 2 arrays for the 2 sets. No more extra arrays are needed.

- ❏ Do not use any goto statements. You may not use any breaks or return statements inside any loops - you are allowed to use breaks inside a switch statement.
- ❏ Must use appropriate loop for reading input. All input data must be validated. For example, you should not accept any characters for numerical data types.
- ❏ Must not use any sorting algorithms other than what is given.
- ❏ Must use all functions mentioned under Task - there should be no code redundancy. **All functions must be implemented after main() and follow above mentioned requirements**.
- ❏ Print a welcome and goodbye message.

- ❏ Please open and compare your work with the grading rubric before submitting.
- ❏ Remember to follow all style guidelines.
- ❏ Download your Algorithmic Design Document as a PDF (File -> Download -> PDF), rename it to `a03.pdf`, and upload it to the D2L assignment by **Wednesday**.
- ❏ Upload your `a03.cpp` C++ source file to the D2L assignment by **Sunday**.
- ❏ Do your own work. Consult the syllabus for more information about academic integrity.

Additional Support

- ❏ Post a question for the instructor in the Ask Questions! area of the Course Lobby.
- ❏ To see how the Selection Card Sort Algorithm works, watch this video from Virginia Tech.
- ❏ Try this Selection Sort Animation by Y. Daniel Liang.
- ❏ Watch this Python Video to help you with the sorting algorithm.