## Project Details

This project involves developing a several statistical methods that are commonly used for analyzing data. The methods developed along with some details are provided below:

* Method: std::vector<**int**> **readVector**(std::istream& is): This method reads integers from the given input stream until -1 is encountered and returns the numbers read stored in a vector. The -1 (which is a sentinel value used to indicate end of list) is not stored in the returned vector.
* Method: **void** **printList**(**const** std::vector<**int**>& list, std::ostream& os): This method Prints each value in the vector list (followed by a space) in the given list to the given output stream os.
* Method: **double** **mean**(**const** std::vector<**int**>& numbers): This method must return the mean of the given list of numbers. If the vector contains the numbers {1, 2, 3, 4}, then the mean is (1+2+3+4)/4 = 2.5.
* Method: **double** **median**(**const** std::vector<**int**>& numbers): This method must return the median value for the given list of numbers. The median is described as the numerical value separating the higher half of the numbers from the lower half. The median of a list of numbers is found by arranging (or sorting) all the observations from lowest value to highest value and picking the middle one. If there is an even number of observations, then there is no single middle value; the median is then usually defined to be the mean of the two middle values. For example, if numbers contains the values {3,4,1,2, 5} then the median is 3 (middle value in the sorted list: {1,2,**3**,4,5}). On the other hand, if numbers contains the values {3,6,4,1,2,5} then the median is 3.5 (average of middle values in the sorted list: {1,2,**3,4**,5,6}).

In order to sort the given list of numbers, make a copy of the given vector using a suitable constructor (as the parameter is constant and cannot be modified) and then sort the numbers using call to std::sort() method as shown below:

|  |
| --- |
| std::vector<**int**> copy(numbers);  std::sort(copy.begin(), copy.end()); |

* Method: **int** **mode**(**const** std::vector<**int**>& numbers): This method must return the mode of the given list of numbers. The mode is the value that most often occurs in the given list of numbers. For example, the mode of {3,2,5,1,2,3,2,4} is 2 (since 2 is the most frequently occurring value). If two or more values have the same frequency of occurrence then any one of the values is valid as the mode. For example, if the list is {1,1,1,2,2,2} then mode can be 1 or 2 (and which one you choose is up to you).
* Method: **bool** **contains**(**const** std::vector<**int**>& list1, **const** std::vector<**int**>& list2): This method returns true only if every number in list2 (immaterial of position in list2) is in list1 (immaterial of position in the list1). For example contains({2,5,3,3,1,2}, {1,3,5}) should return true. On the other hand, contains({2,5,3,3,1,2}, {1,**6**,5}) should return false (because 6 is not present anywhere in list1 in this example).

## Sample Outputs

Outputs from multiple independent runs of the completed program are shown below. User inputs are shown in red for improved readability

|  |
| --- |
| Enter first list of numbers (-1 to end list): 3 2 1 5 4 3 2 2 4 5 -1  Statistics about the first list of numbers:  Numbers: 3 2 1 5 4 3 2 2 4 5  mean = 3.1  median = 3  mode = 2  Enter second list of numbers (-1 to end list): 2 1 3 -1  Statistics about the second list of numbers:  Numbers: 2 1 3  mean = 2  median = 2  mode = 1  Checking to see if one list contains the other:  First list contains second?: yes  Second list contains first?: no |

|  |
| --- |
| Enter first list of numbers (-1 to end list): -3 -5 1 0 10 0 3 0 100 200 300 -1  Statistics about the first list of numbers:  Numbers: -3 -5 1 0 10 0 3 0 100 200 300  mean = 55.0909  median = 1  mode = 0  Enter second list of numbers (-1 to end list): 0 200 300 -1  Statistics about the second list of numbers:  Numbers: 0 200 300  mean = 166.667  median = 200  mode = 0  Checking to see if one list contains the other:  First list contains second?: yes  Second list contains first?: no |

|  |
| --- |
| Enter first list of numbers (-1 to end list): 1 1 3 3 2 2 5 5 4 4 -1  Statistics about the first list of numbers:  Numbers: 1 1 3 3 2 2 5 5 4 4  mean = 3  median = 3  mode = 1  Enter second list of numbers (-1 to end list): 1 4 5 3 2 -1  Statistics about the second list of numbers:  Numbers: 1 4 5 3 2  mean = 3  median = 3  mode = 1  Checking to see if one list contains the other:  First list contains second?: yes  Second list contains first?: yes |