



//Shrabanti Basu

//April 24, 2016

//Program 10

//This program calculates reads data from three files; stores and sorts them in an array,

//prints unsorted and sorted arrays. It saves all the elements from the three arrays into a

//bigger array.

//It also calculates mean, median, and mode for the arrays.

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

//function prototypes

void printArray(double arr[], int);

void mean(double arr[], int);

void sort(double[], int);

void median(double[], int);

void mode(double[], int);

int main()

{

cout << "Shrabanti Basu\n";

cout << "April 24, 2016\n";

cout << "Program 10\n";

cout << "This program calculates reads data from three files; stores and sorts them in an array\n"

<< "print unsorted and sorted arrays. It saves all the elements from the three arrays into a\n"

<< "bigger array. It also calculates mean, median, and mode for the arrays.\n\n";

ifstream inFile; //to store input file stream object

string fileName; //to store name of input file

double numbers[100]; //declare an array of doubles with size 100 to hold individual file elements

double numbersLarge[300]; //declare another array of doubles with size 300 for all elements of three files

double num = 0.0; //to store a number read from the file

string fileInfo; //first line of file information

int total = 0; //accumulator; keeps track of total number of items in the bigger array

int fileNum = 1; //to store the number of files to be read and initialized to 1

while (fileNum <= 3)

{

cout << "Enter file name for file # " << fileNum << " : ";

cin >> fileName;

inFile.open(fileName); //open the input file with user entered filename

if (inFile.fail())

cout << "Error opening File.\n\n";

else

{

//read and print file information

getline(inFile, fileInfo, '\n');

cout << "File information for file # " << fileNum << " : \n";

cout << fileInfo << endl;

int i = 0; //counter; keeps track of array position for each individual array

//read a number from the file and check for the sentinel value before entering it in the array

inFile >> num;

while (num != -999)

{

numbers[i] = num;

i++;

numbersLarge[total] = num; //fill the array larger array

total++;

inFile >> num;

}

inFile.close();

//print the unsorted array

cout << "Unsorted Array\n";

printArray(numbers, i);

//sort the array and print

sort(numbers, i);

cout << "Sorted Array\n";

printArray(numbers, i);

//call functions for mean, median and mode

mean(numbers, i);

median(numbers, i);

mode(numbers, i);

cout << endl << endl;

fileNum++; //increment to go to the next file

}

}

cout << "NOW PRINTING THE BIGGER ARRAY\n\n";

cout << "The elements of the three individual files were all saved in a bigger array.\n";

cout << "The information of the bigger array is as follows:\n";

//print the unsorted bigger array

cout << "Unsorted Array\n";

printArray(numbersLarge, total);

//sort bigger array and print again

sort(numbersLarge, total);

cout << "Sorted Array\n";

printArray(numbersLarge, total);

//call mean, median and mode functions

mean(numbersLarge, total);

median(numbersLarge, total);

mode(numbersLarge, total);

return 0;

}

//function definition

//printArray function prints the elements of the unsorted array

void printArray(double arr[], int size)

{

cout << "There are " << size << " items in this array:\n";

for (int c = 0; c < size; c++)

cout << arr[c] << " ";

cout << endl << endl;

}

//function definition

//sort function uses a bubble sort algorithm to sort elements of an array

//in ascending order.

//size is the number of elements in the array.

//I have used the bubble sort algorithm from the textbook (page 471 - 473)

void sort(double arr[], int size)

{

bool swap;

double temp;

do

{

swap = false;

for (int c = 0; c < (size - 1); c++)

{

if (arr[c] > arr[c + 1])

{

temp = arr[c];

arr[c] = arr[c + 1];

arr[c + 1] = temp;

swap = true;

}

}

} while (swap);

}

//function definition

//mean function calculates and prints the mean / average of elements of an array

//it divides the sum of all array elements and divides it by the array size

void mean(double arr[], int size)

{

double sum = 0.0, mean = 0.0; //to store sum of all elements and their average; initialized to zero

for (int c = 0; c < size; c++)

sum += arr[c];

mean = sum / size;

cout << "The mean is " << mean << endl;

}

//function definition

//the median function calculates and prints the median of the numbers in the array.

//if there are odd number of elements in the array, the median is the middlemost value

//if there are even number of elements in the array, the median is the average of the two middlemost values

void median(double arr[], int size)

{

int quotient = 0, remainder = 0; //to hold the quotient and modulus value when size of array divided by 2

double med = 0.0; //to hold median value

quotient = size / 2;

remainder = size % 2;

if (remainder == 0)

{

med = (arr[quotient - 1] + arr[quotient]) / 2.0;

}

else

{

med = arr[quotient];

}

cout << "The median is " << med << endl;

}

//function definition of mode

//Source: http://www.cplusplus.com/forum/beginner/116489/

//I modified to make it work for an array of doubles,

//added conditions to print a mode value only when there is a mode.

//I also changed some of the initialization values

void mode(double arr[], int size)

{

int counter = 1; //to count occurrences of a number in the array

int max = 1; //another counter to set 1

double mode = arr[0]; //to store the mode value; initialized to the first array element

//iterate through the array to check if the contiguous elements are same.

//we use this process because the array is already sorted

//if the elements are same, increment the counter and save the value in mode

//if the number is not repeated, set the counter back to 1

for (int pass = 0; pass < size - 1; pass++)

{

if (arr[pass] == arr[pass + 1])

{

counter++;

if (counter > max)

{

max = counter;

mode = arr[pass];

}

}

else

counter = 1; // reset counter.

}

//when the value of max is greater than 1 there is a mode

//print the mode or print appropriate message

if (max > 1)

cout << "The mode is: " << mode << endl << endl;

else

cout << "No mode\n\n";

}