// Joshua Saunders

// CO SCI 140 (Section 15029) - Assignment 01

// Due February 17, 2019

//

// Prompt:

// Write a program that dynamically allocates an array large enough to hold a

// user-defined number of test scores. Once all the scores are entered, the

// array should be passed to a function that sorts them in ascending order.

// Another function should be called that calculates the average score. The

// program should display the sorted list of scores and averages with

// appropriate headings. Use pointer notation rather than array notation

// whenever possible.

//

// Input Validation: Do not accept negative numbers for test scores.

//

// 10,000 ft. view:

// 1. Ask user how many grades they want to enter

// a. Validate input: make sure that the number of grades >= 0

// 2. Create an array that's the same size as the number of grades to be

// entered

// a. Validate input: make sure that grades are only >= 0

// 3. Sort the array in ascending order

// 4. Find the average of the test scores

// 5. Display the test scores (sorted in ascending order) and average score

// a. Only show two decimal places

#include <iostream>

#include <iomanip>

// Function prototypes

double calcAverage**(**const double **\*,** int**);**

void displayResults**(**const double **\*,** int**,** double**);**

void sortAscending**(**double **\*,** int**);**

bool isNegative**(**double**);**

int main**()**

**{**

double **\***scores **=** **nullptr;**

int numScores **=** 0**;**

// Get the number of scores from the user, making sure to

// not allow a negative number of scores

std**::**cout **<<** "\nHow many test scores will you enter? "**;**

**do**

**{**

std**::**cin **>>** numScores**;**

**if** **(**isNegative**(**numScores**))**

**{**

std**::**cout **<<** "The number cannot be negative.\nEnter another number: "**;**

**}**

**}** **while** **(**numScores **<** 0**);**

scores **=** **new** double**[**numScores**];**

int scoresCount **=** 0**;**

double score **=** 0.0**;**

// Get the scores from the user, making sure that no negative

// scores are entered

**while** **(**scoresCount **<** numScores**)**

**{**

std**::**cout **<<** "Enter test score " **<<** scoresCount **+** 1 **<<** ": "**;**

std**::**cin **>>** score**;**

**while** **(**isNegative**(**score**))**

**{**

std**::**cout **<<** "Negative scores are not allowed.\n"

**<<** "Enter another score for this test: "**;**

std**::**cin **>>** score**;**

**}**

**\*(**scores **+** scoresCount**)** **=** score**;**

scoresCount**++;**

**}**

// Sort the scores array (in ascending order), find the average

// score, then display the results

sortAscending**(**scores**,** numScores**);**

double averageScore **=** calcAverage**(**scores**,** numScores**);**

displayResults**(**scores**,** numScores**,** averageScore**);**

// Don't forget to free up the dynamically allocated memory and

// set the pointer to the null pointer

**delete** **[]** scores**;**

scores **=** **nullptr;**

**return** 0**;**

**}**

double calcAverage**(**const double **\***arr**,** int size**)**

**{**

// Returns the average of the

double total **=** 0.0**;**

**for** **(**int i **=** 0**;** i **<** size**;** i**++)**

**{**

total **+=** **\*(**arr **+** i**);**

**}**

**return** total **/** **((**double**)** size**);**

**}**

void displayResults**(**const double **\***arr**,** int size**,** double average**)**

**{**

// Displays the array (arr) and the average (average) as follows:

//

// The test scores in ascending order, and their average, are:

//

// Score

// -----

//

// 82.30

// 84.70

// 97.50

//

// Average score: 88.17

std**::**cout **<<** std**::**fixed **<<** std**::**showpoint **<<** std**::**setprecision**(**2**);**

std**::**cout **<<** "\nThe test scores in ascending order, and their average, are:\n\n"

**<<** " Score\n -----\n\n"**;**

**for** **(**int i **=** 0**;** i **<** size**;** i**++)**

**{**

std**::**cout **<<** " " **<<** **\*(**arr **+** i**)** **<<** std**::**endl**;**

**}**

std**::**cout **<<** "\nAverage score: " **<<** average**;**

**}**

void sortAscending**(**double **\***arr**,** int size**)**

**{**

// Implements the selection sort algorithm to sort arr

// in ascending order (chapter 8, section 3, pg 474-475)

int startScan**,** minIndex**;**

double minValue**;**

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

**{**

minIndex **=** startScan**;**

minValue **=** **\*(**arr **+** startScan**);**

**for** **(**int index **=** startScan **+** 1**;** index **<** size**;** index**++)**

**{**

**if** **(\*(**arr **+** index**)** **<** minValue**)**

**{**

minValue **=** **\*(**arr **+** index**);**

minIndex **=** index**;**

**}**

**}**

**\*(**arr **+** minIndex**)** **=** **\*(**arr **+** startScan**);**

**\*(**arr **+** startScan**)** **=** minValue**;**

**}**

**}**

bool isNegative**(**double score**)**

**{**

// Returns true if negative, returns false otherwise

**return** score **<** 0**;**

**}**