COMP-165-0837

Merlin Newcomb

Pointer Assignment

Problem #1

Code:

/\*

\* 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 accepts negative numbers from test scores.

\*/

#include <iostream>

#include <algorithm>

#include <numeric>

// Returns the average of a C array of ints, rounded down.

int getAverage(int\* userScores, int size);

int main(void)

{

int userLen = 0;

std::cout << "Enter number of test scores: " << std::flush;

std::cin >> userLen;

int\* userScores = new int[userLen];

for (int i = 0; i < userLen; i++)

{

int userInput = 0;

while (true)

{

std::cout << "Score #" << (i + 1) << ": " << std::flush;

std::cin >> userInput;

if (userInput >= 0)

{

userScores[i] = userInput;

break;

}

std::cout << "Invalid score. Please input non-negative integer." << std::endl;

}

}

std::sort(userScores, userScores + userLen); // An STL function that sorts an array.

int userAvg = getAverage(userScores, userLen);

for (int i = 0; i < userLen; i++)

{

std::cout << userScores[i] << " ";

}

std::cout << "\n" << std::endl;

std::cout << "Average of scores: " << userAvg << std::endl;

delete userScores;

return 0;

}

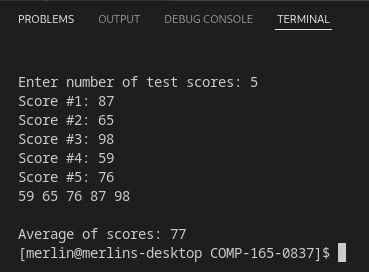
int getAverage(int\* userScores, int size)

{

return std::reduce(userScores, userScores + size) / size;

}

Output:



Problem #2

Code:

/\*

\* Write a function that accepts an int array and the array's size as arguments.

\* The function should create a new array that is twice the size of the argument array.

\* The function should copy the contents of the argument array to the new array and initialize the unused elments of the second array with 0.

\* The function should return a pointer to the new array.

\*/

#include <iostream>

// Doubles passed array's size, fills the rest of the array with 0.

int\* continueArray(int\* array, int size);

// Creates array and fills based on a base number and increments, prints out original and doubled array.

int main(void)

{

int baseNumber = 0;

int incrementNumber = 0;

std::cout << "Enter base number: " << std::flush;

std::cin >> baseNumber;

std::cout << "Enter increment number: " << std::flush;

std::cin >> incrementNumber;

int firstArray[5];

for (int i = 0; i < 5; i++)

{

firstArray[i] = (baseNumber + (incrementNumber \* i));

std::cout << firstArray[i] << ", ";

}

std::cout << std::endl << std::endl;

int\* newArray = continueArray(firstArray, 5);

for (int i = 0; i < 10; i++)

{

std::cout << newArray[i] << ", ";

}

std::cout << std::endl;

return 0;

}

int\* continueArray(int\* array, int size)

{

int newArrayLength = size \* 2;

int\* newArray = new int[newArrayLength];

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

{

newArray[i] = array[i];

}

for (int i = size; i < newArrayLength; i++)

{

newArray[i] = 0;

}

return newArray;

}

Output:

