|  |
| --- |
| Homework : \_\_\_4.A\_\_\_\_\_\_  4.A Source code  //Christopher Badolato CH432391  //Assignment 4A  //10/10/2018  //ENC3211  //This program will calculate standard deviation, adjusted the exam scores  //based on the deviation, then sort the exam scores  #include <stdio.h>  #include <math.h>  int main() {  //loops and temp values  //we need a few arrays for this assignment, our existing array  //a copy to sort, one to hold the sums of our deviations, and one for  //our adjusted exam scores, also our final sorted adjusted scores.  //along with all of our float variables needed.  int examScores[] = {80, 82, 90, 95, 90, 87, 92};  int sortedArray[] = {80, 82, 90, 95, 90, 87, 92};  float valuesMinusAvergeSquared[7];  float adjustedExamScores[7];  float sortedAdjustedExamScores[7];  int i, j, temp;  float standardDeviation, average, sumOfValues = 0, sumOfValuesMinusAverageSquared;  float valueUnderRoot, oneSeventh;  //print our exam scores  printf("Exam Scores: ");  for(i = 0; i < 7; i++){  printf(" %d", examScores[i]);  }  //first step of finding standard deviation is to find the average;  printf("\n");  printf("\n");  for(i = 0; i < 7; i++){  valuesMinusAvergeSquared[i] = 0;  sumOfValues = sumOfValues + examScores[i];  }  average = sumOfValues/7;  //after we find the average we need to subtract each value from  //the average and square them and sum them all together.  for(i = 0; i < 7; i++){  valuesMinusAvergeSquared[i] = pow((float)examScores[i] - average, 2);  sumOfValuesMinusAverageSquared = sumOfValuesMinusAverageSquared + valuesMinusAvergeSquared[i];  }  //get the final value underneath the root by multiplying by n (7)  //then take square root to get the std deviation.  valueUnderRoot = sumOfValuesMinusAverageSquared/7;  standardDeviation = sqrt(valueUnderRoot);  printf("Standard deviation: %.04f\n", standardDeviation);  printf("\n");  //Here we will adjust the exams scores by the standard deviation then  //print the list back out.  printf("Adjusted exam scores: ");  for(i = 0; i < 7; i++){  adjustedExamScores[i] = examScores[i] + standardDeviation;  printf(" %.0f", adjustedExamScores[i]);  }  printf("\n");  printf("\n");  /\*  To sort we much start by going through each number at least once  we will go through again starting at our first term checking to  see if it is greater than the value next to it.  if it is, swap the values on the list.  To make sure the last elements are already in place  on the inside loop we subtract i from the length of the array.  therefore each pass it makes one less comparison and we can print  as we get those values.  \*/  for (i = 0; i < 7; i++){  for (j = 0; j < 7-i-1; j++)  if (sortedArray[j] > sortedArray[j+1]){  temp = sortedArray[j];  sortedArray[j] = sortedArray[j+1];  sortedArray[j+1] = temp;  }  }  printf("Sorted exam scores: ");  //finally we adjust the sorted scores  //and reprint the adjusted list.  for(i = 0; i < 7; i++){  sortedAdjustedExamScores[i] = sortedArray[i] + standardDeviation;  printf(" %.0f", sortedAdjustedExamScores[i]);  }  printf("\n");  return 0;  } |
|  |
| 1.A Program output  Exam Scores: 80 82 90 95 90 87 92  Standard deviation: 4.9857  Adjusted exam scores: 85 87 95 100 95 92 97  Sorted exam scores: 85 87 92 95 95 97 100  Process returned 0 (0x0) execution time : 0.040 s  Press any key to continue. |
| Homework : \_\_4.B\_\_\_\_\_\_\_  4.B Source Code  //Christopher Badolato CH432391  //Assignment 4B  //9/30/2018  //This assignment will check the arrays to see if they  //are a palindrome  #include <stdio.h>  //prototypes for each function, we will reverse the array  //compare it to the original and then print the original  //along with whether or not it is a palindrome.  void createReserveArray(int arrayA[], int size);  int compareArray(int arrayA[], int arrayB[], int size);  void printArray(int arrayA[], int size, int palindromeCheck);  int main() {  int myArray1[]={1,2,3,2,1};  int myArray2[]={1,2,3,4,1};  int myArray3[]={1,2,3,3,2,1};  int myArray4[]={1,2,3,4,2,1};  //Call our functions on each of the arrays  //we also send the size of the array.  createReserveArray(myArray1, 5);  createReserveArray(myArray2, 5);  createReserveArray(myArray3, 6);  createReserveArray(myArray4, 6);  return 0;  }  void createReserveArray(int arrayA[], int size){  //We need a new array to store the reverse in  //and I also create a copy of the original to send  //to our check functions.  int tempArray[size];  int originalArray[size];  int i;  int palindromeCheck = -1;  for(i = 0; i < size; i++){  originalArray[i] = arrayA[i];  //will make sure the array is reversed  tempArray[i] = arrayA[size-i-1];  }  //call our check with the originalArray, reversed array, and the size.  //then call our print function  //we also create an integer value for our palidromeCheck.  palindromeCheck = compareArray(originalArray, tempArray, size);  printArray(originalArray, size, palindromeCheck);  }  int compareArray(int arrayA[], int arrayB[], int size){  int i, j;  /\*in this function we are comparing each value of the arrays  (original and reversed) at the same index  if the values are the same return 0; if different return 1;  if the values are the same keep looping  if they are different we can say that its not a palindrome return 1.  \*/  for(i = 0; i < size; i++) {  if(arrayA[i] == arrayB[i]) {  continue;  }  else {  return 1;  }  }  return 0;  }  void printArray(int arrayA[], int size, int palindromeCheck){  //Finally we will print our array, and use our check value  //to see if it is a palindrome or not.  int i;  printf("Array elements are: ");  for(i = 0; i < size; i++){  printf("%d ", arrayA[i]);  }  printf(" \n");  if(palindromeCheck != 0){  printf("The array is not a palindrome");  }  else{  printf("The array is a palindrome");  }  printf(" \n \n ");  } |
|  |
| 4.B Program output  Array elements are: 1 2 3 2 1  The array is a palindrome  Array elements are: 1 2 3 4 1  The array is not a palindrome  Array elements are: 1 2 3 3 2 1  The array is a palindrome  Array elements are: 1 2 3 4 2 1  The array is not a palindrome  Process returned 0 (0x0) execution time : 0.047 s  Press any key to continue. |