Programming Assignment – Programming in C - Arrays

1. Create a C program file named ***arrays.cc*** that contains the methods from steps 2 through 6 below.

   DO NOT USE ANY CONSTANTS FROM MAIN IN YOUR FUNCTIONS. YOU MUST USE ONLY ARRAY SIZES PASSED AS PARAMETERS OR YOU WILL FAIL ALL OF THE TESTS AND WILL RECEIVE A ZERO FOR A GRADE. YOU HAVE BEEN WARNED.

   YOU MAY ONLY CREATE THE FUNCTIONS LISTED BELOW. NO HELPER FUNCTIONS ALLOWED.

   YOU ARE ONLY ALLOWED TO USE STDIO.H

2. Create a method named ***getPositiveAverage*** which takes a double array and the number of items in the array as parameters. This method should return the average of all of the positive items in the array as a double. If the array is empty or there are no positive values, zero should be returned.

   Here is what the function header should look like. The parameter names do not matter.

   ```
   double getPositiveAverage(double myArray[], int numItems)

   {

   }
   ```

3. Create a method named ***countRangeValues*** which takes a double array, the number of items in the array (int), and a value to count (double) as parameters. This function should count and return (int) how many times the argument in the third parameter occurs in the array within the range of ± 0.5.

   That is, if the third argument, x, is 10.0 the range of values to count would be **9.5 ≤ x < 10.5**

   for the array {1.1, 1.2, 1.5, 1.8**, 2.0, 2.1, 2.2, 2.4, 2.7**, 3.0, 3.1, 3.2, 3.5, 3.7}

   countRangeValues(myArray, 14, 2.5);

   should count the bold values above in the range of 2.0 to 2.9999999… and return 5.

   NOTE: In the above example, 2.0 should be counted but 3.0 should not.

4. Create a method named ***getMaxAbsolute*** which takes a double array and the number of items in the array as parameters. This method should return the item from the array with the largest absolute value. Return the ***ITEM*** and ***NOT*** the absolute value.

   That is if the array is { -2.5, -10.1, 5.2, 7.0}, your function should return -10.1

   and if the array is {5.1, 2.3, 4.9, 1.0}, your function should return 5.1.

   If there is a negative and a positive of the same maximum magnitude, return the positive value.

5. Create a method named ***countInverses*** which takes an int array and the number of items in the array and returns the number of positive value / additive inverse pairs in the array.

   For example:

   The following array {-2, -6, 4, 5, 2, 8, 9, -8, -4, 3}
   has the following positive value/additive inverse pairs: 2/-2, 4/-4, 8/-8

   So the return value should be 3 since there are 3 pairs.

   Any duplicates should be counted only as many times as there are pairs.

   The following array {-1, 1, 1, 1, -1, 1, 3, -3, 3, 3, 3, 1, 1, 1, -1, -3}

   should return five since there are five pairs {-1, 1, 1, 1, -1, 1, 3, -3, 3, 3, 3, 1, 1, 1, -1, -3}

   Note that zero IS NOT positive and should not be counted even though it is its own additive inverse.

6. Create a method named ***getMaxCount*** which takes a double array and the number of items in the array as parameters. This method should get the item with the biggest absolute value (get by using a previously created method) and then return the number of times that item occurs (± 0.5) (get by using a previously created method). If the array is empty, zero should be returned.

   YOU ARE NOT ALLOWED TO USE A LOOP IN THIS METHOD! YOU MUST USE METHODS PREVIOUSLY CREATED IN THIS ASSIGNMENT TO GET CREDIT FOR THIS ONE. (Yes! The methods you call can have loops. Loops are just NOT allowed in this function.)

   For example:

   The following array:
   {-5.3, -4.4, -5.5, 1.0, -5.1, 5.2, 1.0, -5.0, 1.0}

   Find the item with the highest absolute value (-5.5) and return the number of times that something near that value (within ± 0.5) occurs in the array (3 times -5.2, -5.5, and -5.1 but not -5.0).

7. Use a main.cc file with a main method for testing your program before uploading. There is a main.cc on AsULearn with some tests.

   MAKE YOUR OWN TESTS WITH DIFFERENT ARRAYS!!! Make sure to test with negative elements and zeros.

   DO NOT UPLOAD MAIN.CC OR ANY FILE WITH A MAIN METHOD IN IT!

   DO NOT PUT A MAIN FUNCTION IN arrays.cc (NOT EVEN FOR TESTING)

   Do it the correct way. If you don't understand **ASK**!!!

The command for compiling is the same as last week but with arrays.cc.

```
g++ -Werror –Wall -o arrays main.cc arrays.cc
```

Run using ***arrays*** or ***./arrays***

You will get an error message on test fail in main. If nothing is printed you passed all tests.

8. DO NOT PUT A MAIN METHOD INSIDE OF arrays.cc. A main method should ONLY be in main.cc. If you cannot figure out how to make your program compile without putting a main method in arrays.cc, come see me.

   YOUR TESTS WILL FAIL WHEN SUBMITTED TO WEB-CAT IF YOU PUT A MAIN FUNCTION INSIDE OF arrays.cc.

9. Submit your ***arrays.cc*** file to Web-CAT for grading.

   Do not submit a file with a main function.
   DO NOT submit main.cc.
   File cannot be submitted more than 2 days late.
   Late submissions will suffer a 10 point penalty per day.
   You will lose 1 point for each submission over 5.