

ASSIGNMENT 8

20 points

CSC 140 Spring 2014
Date assigned: Wednesday, 4/30
Date due: Wednesday, 5/7 (1 week)

This assignment involves fundamental array concepts with simple types. The objectives involve demonstrating how to define an array, process array elements, and pass arrays to functions. Using loops with an array also will be an objective.

Objectives to be met (Basis for assignment “points”):	
1	Define 2 arrays, type string and type int, inside your program’s main function.
2	Demonstrate how to access and process array elements safely and consistently with regard to the array boundaries when using loop structures.
3	Demonstrate how the same program will work with a partially or completely filled array.
4	Demonstrate how to define and call 2 different functions with array parameter types.

Your program should obtain the vote counts for candidates running for election (there will not be more than 5 candidates, but there could be less). Initialize the candidates array with names, and the vote counts should be obtained from the user (keyboard) and stored in the votes array while being processed in a loop using the DO-WHILE loop style.

After the first input loop ends, your program should use a second (FOR) loop to display the list of only those candidates who received 0 or more votes. Output should consist of each candidate’s name, vote count, and percentage of total votes received. That means before printing the list, you must get the total vote count for all candidates – the percentage for each candidate will be...

candidate votes / total votes * 100

Since the votes array is an integer and the total also should be an integer, you will need to use type casts to get “real division.”

Run the program twice using an example with just 3 candidates and then again using all 5 candidates. (See sample output requirements included with these instructions.)

Restrictions: (1) Do NOT prompt the user for the number of input values – your input loop must ask the user whether or not to continue. (2) Do NOT under any circumstances use a `break`, `continue`, or `return` statement inside of any loop – All loops must exit through their respective conditional expressions. (3) Do NOT use global variables in your program unless they are defined and initialized as a constant.

Minimum requirements for a “letter grade of C” (also see Recommended Improvements):	
1	Adhere to all 3 Restrictions stated above.
2	Your program must define 2 arrays that can store exactly 5 values: candidates of type string and votes of type int.
3	Your program must demonstrate processing the input using the DO-WHILE loop and other processing using a FOR loop.
4	Your main function must call and pass the both arrays and the count to a function implementing the Print List module, and that function must call the function implementing the Get Total module. (See Functional Diagram at the end of these instructions.)
5	Run the program twice and provide 2 examples of program output: one using only 3 input values (partially filled array) and the other using 5 input values (completely filled array).

Programming style requirements regarding functions:

- Always declare prototypes for every function at the top of the program listing before main()
- Always define functions at the bottom of the program listing after main()
- Always include names (identifiers) in function prototypes for every parameter
- All functions must return only at the end of their statement block; do not use a return statement or an exit() inside decision or loop statements (or anywhere else) that might cause the function to end early.

Basic Minimum Requirement Program Interaction:

```
Enter number of votes for Johnson: 5000
Another (y or n): y
Enter number of votes for Miller: 4000
Another (y or n): y
Enter number of votes for Doddy: 6000
Another (y or n): n
```

Candidate	Votes	% of Total
-----	-----	-----
Johnson	5000	33.33%
Miller	4000	26.67%
Doddy	6000	40.00%
-----	-----	-----
	15000	

The WINNER is: Doddy

End program.

```
Enter number of votes for Johnson: 5000
Another (y or n): y
Enter number of votes for Miller: 4000
Another (y or n): y
Enter number of votes for Doddy: 6000
Another (y or n): y
Enter number of votes for Maxwell: 6050
Another (y or n): y
Enter number of votes for Bishop: 2000
```

That's all the candidates.

Candidate	Votes	% of Total
-----	-----	-----
Johnson	5000	21.69%
Miller	4000	17.35%
Doddy	6000	26.03%
Maxwell	6050	26.25%
Bishop	2000	8.68%
-----	-----	-----
	23050	

The WINNER is: Maxwell

End program.

Considerations:

- Be careful with array subscripts in your loops. Your input loop, especially, will need to check 2 exit

conditions: the user response and the array size. Keep count of the values being entered so that remaining loop(s) only need to check that count, which should be less than or equal to the array size if your input loop is designed properly.

- Be sure to review examples in our text book, classroom examples, and lab exercise about passing an array along with other arguments to another function.
- If you write a function for the Get Votes module (Recommendation #3) then give careful consideration to that module in the functional diagram at the end of these instructions. There are two values returned but one of them is the votes array. This should be declared as a value-returning function with a return statement returning the count, but the array returning the votes is a parameter – you cannot return an array with a return statement. That function must keep count of how many values were input and send the count back to main – other functions will need that count. The rest of the program should process only the count for those candidates who received 0 or more votes.

Recommended “Letter Grade” Improvements:

(1) Use decision logic inside your input loop to process 0 and positive input values as valid votes; Negative input values should NOT be processed as valid votes but also should NOT cause the loop to exit; Instead it should simply display a message about not accepting negative input and allow the user to exit or enter another value for that candidate. (You can assume all vote amounts that are input will be numeric values.)

Example with negative input value:

```
Enter number of votes for Johnson: 5000
Another (y or n): y
Enter number of votes for Miller: 4000
Another (y or n): y
Enter number of votes for Doddy: -1
Votes receieved cannot be negative.
Another (y or n): y
Enter number of votes for Doddy: 0
Another (y or n): y
Enter number of votes for Maxwell: 6050
Another (y or n): n
```

Candidate	Votes	% of Total
Johnson	5000	33.22%
Miller	4000	26.58%
Doddy	0	0.00%
Maxwell	6050	40.20%
	15050	

The WINNER is: Maxwell

End program.

(2) Declare and write a function for the Get Index of Winner module. This is a value-returning function with a return statement that returns the index location of the votes array with the largest value, and takes two incoming (read-only) parameters including the votes array and the count of how many candidates received votes. (To keep this simple, assume there are no ties – no two candidates have the same vote count.)

(3) Declare and write a function for the Get Votes module. This is another value-returning function with a return statement that returns the count of how many candidates received 0 or more votes. There are 3

parameters that include the candidates array, the votes array, and the size of the arrays. The candidates array and the size both are incoming (read-only) parameters and the votes array is an outgoing parameter.

Additional Requirements:

Include your name, course, section (CSC140-xxx) at the top of your program file as a comment. Compress and submit your project folder as 1 .zip file. (Your .cpp file must be inside the project folder.)

Level 0 Solution:

Define and initialize constant array size
Define and initialize constant candidates array
Define votes array

Get Votes

Print List

Get Total Votes

Get Index of Winner

Print name of candidate who won

2 Shaded Modules Required:

Print List, Get Total Votes

Recommendation #2:

Get Index of Winner

Recommendation #3:

Get Votes

Functional Decomposition

