CMSC 257: Assignment 2
C Programming Basics

Summary:

Systems programmers make extensive use of many low level languages, such as C. In this course, you will be using C for many different systems programming tasks. In this assignment, you will create a C program to display your knowledge of the basics of C programming. To get you started, you are provided with a few files in a tarball.

Deliverable:

A g-zipped tarball called "**asgn2_[your_eID].tgz**" (without quotes, replace your_eID with your VCU eID). In this archive should be the files "**asgn2.c**", "**asgn2_support.c**", "**asgn2_support.h**", and "**Makefile**". The purpose of these files is as follows:

1) asgn2.c – The main file. This file contains the main method, which will call the support functions as needed to complete the assignment. The beginning of this file is provided.
2) asgn2_support.c – The support file. This file contains the support functions, as defined later in this document. This file is not provided.
3) asgn2_support.h – The header file. This file allows the compiler to link the two C source files. This file is provided, but must be modified.
4) Makefile – A simple script to compile the project. This file is provided.

Execution Order:

When compiled and executed, your program will do the following:

1) Read in 20 float values from the user, one per line, and place them in an array. The code to perform this is provided.
2) Create a second array of 20 integers. Populate this array by converting the floats to ints as follows:
   a) Truncate the float to an integer
   b) Convert the int to a positive value by taking it's absolute value
   c) Convert the int to the range 0 to 15 inclusive by using the modulus operation
3) Print the contents of the float array to the terminal.
4) Print the contents of the integer array to the terminal.
5) For each integer, print the number of "1" bits in it's binary representation.
6) Sort the integer array using the quicksort method.
7) Print the sorted integer array to the terminal.
8) Print the number of even values in the float array.
9) Print the number of even values in the integer array.
10) Print the most frequent value(s) in the integer array.
11) For each integer in the sorted integer array:
    a) Cast the int to an unsigned short int.
    b) Print the binary representation of this unsigned short int.
    c) Reverse the bits of this binary representation.
    d) Print the reversed binary representation.
    e) Print the value of the reversed binary representation. (Convert it back to decimal).

Extra Credit:

Extra credit will be given for the following:

1) Using bit wise operators to take the absolute values.
2) Using bit wise operators to do the modulus operation.
3) Using bit wise operators to traverse the bits of the int in the countBits function.
4) Using bit wise operators to count the number of even numbers.
5) Swapping values without a temporary variable in the quicksort function.

Support Functions:

The following table outlines the required support functions, their inputs, and their purpose. You may create additional support functions, but you may need to add them to the header file in order to compile and call them properly.

Do NOT use functions from the math.h library. Using this library will result in points off your grade.

Do NOT create additional files, or rename existing files. This will cause the Makefile to cease working and will result in points off of your grade

Do NOT submit additional files or directories in your tarball. Improper tarballs will result in points off your grade.

Comments and formatting ARE a part of your grade. Sample comments are provided in the provided code.

```
void float_display_array(float f_array[], int NUMBER_ENTRIES);
void integer_display_array(int i_array[], int NUMBER_ENTRIES);
int float_evens(float f_array, int NUMBER_ENTRIES);
int integer_evens(int i_array, int NUMBER_ENTRIES);
int count_bits(int a);
void integer_quicksort(int i_array[], int low, int high);
int partition(int i_array[], int low, int high);
void most_values(int i_array[], int NUMBER_ENTRIES, int max);
unsigned short reverse_bits(unsigned short int rev);
void binary_string(unsigned short int a, char arr[], int NUMBER_ENTRIES);
```

Function Name	Input Parameters	Description
float_display_array	A reference to a float array. The length of the array as an int.	Prints the values of an array of floats to the terminal on a single line. Delimit these values with a single space. Only print the first two values to the right of the decimal point.
integer_display_array	A reference to an int array. The length of the array as an int.	Prints the values of an array of ints to the terminal on a single line. Delimit these values with a single space.

float_evens	A reference to a float array. The length of the array as an int.	Return the number of even floats in the array. Ignore anything to the right of the decimal point.
integer_evens	A reference to an int array. The length of the array as an int.	Return the number of even ints in the array.
count_bits	An int	Return the number of “1” bits in the binary representation of the given int.
integer_quicksort	A reference to an int array. The left index of the array (the position to start sorting from). The right index of the array (the position to sort to).	The function should sort the integer array from least to greatest using the quicksort method of sorting. You may use the pseudocode available on Wikipedia.
most_values	A reference to an int array. The length of the array as an int. The highest value possible as an int.	The function should return the value(s) which occur the most in the array. If multiple values occur the most, print the list of these values on a single line delimited by a single space.
reverse_bits	An unsigned short int	Return the unsigned short int that results from reversing the bits of the input unsigned short int.
binary_string	An unsigned short int A reference to a char array The length of the char array as an int.	Populate the char array with the binary representation of the input unsigned int. (Don’t forget to add the string terminator to the char array).