**Homework 1019: Separate Compilation**

Last modified: 8 October 2021

Due: Tuesday, 19 October, noon.

**Homework Assignment**

This assignment must be completed entirely in a Unix environment, either Linux or Mac OS.

1. Fix everything that I indicated is a problem on your 1003 program.
2. Create a file named revbits.h that contains all the definitions and declarations needed to provide the revbits functions described below.
3. Create a file named revbits.c that contains the implementation of two functions:  
   uint8\_t revbits8(uint8\_t value) and uint16\_t revbits16(uint16\_t value)  
   that perform exactly the same operation as described in assignment 1003, except the latter on 16-bit values. You must build up your reversed bit value one bit at a time, using obvious, easily understood code. Use a while loop, but not a for loop.  
     
   You may not use a construct such as

value = (value ^ 0xF0) >> 4 | (value ^ 0x0F) << 4;

value = (value ^ 0xCC) >> 2 | (value ^ 0x33) << 2;

value = (value ^ 0xAA) >> 1 | (value ^ 0x55) << 1;

This is not an assignment in google searching, it is an assignment for you to learn to use the bitwise operators in C.

1. Create a file named atoh.h that contains all the definitions and declarations needed to provide the atoh functions described below.
2. Create a file named atoh.c that contains the implementation of two functions:  
   uint8\_t atoh8(const char\* string) and uint16\_t atoh16(const char\* string)  
   that each convert a hex-value string into an integer value. Both functions should accept a string in the form 0xnn where nn is up to 2 hex digits (for atoh8) or up to 4 hex digits (for atoh16). The digits may each be either upper or lower case. You may not use strtoul or any other string library function; you must do the conversion directly in your own code. If the string does not start with “0x”, or if any of the nn characters are not digits or upper or lower case a through f, your function should return 0.
3. The main function (which is not part of this assignment) must be contained in a file named main.c and should be able to have a line such as  
   #include "revbits.h"  
   and then with no further code be able to call either revbits() function, and similarly for the atoh() functions.
4. Create a file named Makefile that contains the rules and dependencies for your system. You must have the default rule with a target of all whose dependent list consists of revbits, and revbits must be the name of the executable that your Makefile creates. As noted above, the main function must be contained in a file named main.c.

A premium is placed on efficient, understandable, elegant code, with minimal duplication or redundant code.

Make sure your C code adheres to the [coding style guide](https://borax.truman.edu/250/coding_rules.html). In particular, note 2 space indents, brace formatting, no tab characters, and the line length limit. All your code must be valid C89, and you should have no compiler warnings or errors.

Submit the files atoh.h, atoh.c, revbits.h, revbits.c, and Makefile to the [homework submission](https://borax.truman.edu/250/submit.php) page.