Pass By Reference

This exercise will give you a chance to work with pass-by-reference, one of the first, useful things we can do with pointers. On the course homepage you will find a partial implementation called passByReference.c. You can also download a copy of

curl -O https://www.csc2.ncsu.edu/courses/csc230/exercise/exercise08/passByReference.c
curl -O https://www.csc2.ncsu.edu/courses/csc230/exercise/exercise08/expected.txt

The program won't compile until you add a few missing functions. Once it's working, it should print the following output when run:

a = 100 b = 50 c = 25
a = 110 b = 60 c = 35
a = 60 b = 35 c = 110
a = 60 b = 35 c = 109

From the source code, you'll see you have to add three functions. You can't modify the contents of main() at all, but by taking parameters passed by address (and maybe some passed by value also), your three functions will be able to change the contents of variables declared in main.

Here's what your functions need to do:

- **incrementAll( )** : This function will take pointers to a, b and c and a 4th integer parameter. Its job is to increment a, b and c by the value of the 4th parameter. The main( ) function uses this to add 10 to each of the variables.
- **rotate( )** : This function will take pointers to a, b and c. Its job is to simultaneously copy the value of b to a, c to b and a to c. You'll probably want to use a temporary, local variable in the rotate() function to hold the integer value one of the parameters points to while you're moving values around.
- **getLargest( )** : This function will take the addresses of a, b and c and it will return the address of the one containing the largest value. So, you'll be creating a function that returns a pointer.

  We looked at an example of return-by-address in class. You need to give **int *** as your function's return type (to return a pointer to int). Inside your function, you'll need to compare the values your three parameters point to, and figure out which is the largest. Then, you'll need to return a copy of the pointer that points to the largest one. Since your parameters are already pointers to ints, you can just return a copy one of your parameters. If you find yourself trying to use the address-of operator here, you're probably making a mistake.

Your program should guarantee that the functions will not modify the pointers (e.g., the pointer to a should not be modified to point to b instead) that are passed to them (hint: use the const qualifier to ensure this).

When you're done submit your completed passByReference.c file to the exercise_08 assignment in Moodle.