## Assignment #5 - Repeated Squaring

### Assignment Due Date, Time, and Format

This assignment has both an electronic submission part and a paper submission part. The electronic submission part will be your source code file. The source code is due as listed in Canvas, before midnight. Submit a single C++ source code file by way of the COSC 4553 Assignment #5 link on Canvas. Do not submit any other files.

Your finished source code, when combined with the provided source code, should compile, link and run with no error messages. It should also run correctly when given expected input.

The paper submission part will be problems for you to solve, and will be given to you by the instructor. The problems will involve RSA key generation, which requires modular exponentiation. Your answers to the problems will be due at the start of class on Tuesday, February 18th. The instructor will give more details about this part of the assignment in class.

To get full credit for this assignment, you must submit both a correct implementation of the repeated squares algorithm and correct answers for the RSA key generation problems.

This is an individual or partner assignment. Do not exchange your solution with other students/partnerships. Both partners should submit the source code file to Canvas. For the paper part of the assignment, each partner should figure out and submit his/her own separate answers to the problems.

### Assignment Objectives

* Practice implementing the repeated squaring algorithm for performing modular exponentiation
* Practice using the implementation to solve mathematical problems that involve modular exponentiation

### Assignment Summary

In this assignment you will complete the definition of two functions that implement the repeated squaring algorithm described in the Stamp textbook on pages 98-99. Note that your implementation should not use the pow() function, the powl() functions, or any other built-in exponentiation functions.

### Assignment Directions

1. Download each of these files from the Assignment #6 page on Canvas:
   * **program6-driver.cpp** - This file contains a completed main() function that serves as the driver to parse the command line, pass the values to the powerModN() function, and print the result. Make no changes to this file.
   * **powerUtility.h** - This file contains the function prototype for the powerModN() function. Make no changes to this file.
   * **student6.cpp** - This file contains two stubbed-out functions. This is the file where you will put your source code additions. This is also the file that you will submit to Canvas.
2. Change the name of the **student6.cpp** file to **Name6.cpp**, where "Name" is your last name. Also, complete the remaining fields in the comment block at the top of the file.
3. Make sure that the source code files downloaded from Canvas will compile, link, and run with no errors by entering the following at the command shell prompt:

% g++ program6-driver.cpp Name6.cpp  
% a.exe

A program usage message should appear on the screen, and the program should exit.

1. Finish the implementation of the two stubbed-out functions in the Name6.cpp file. (Add no other function implementations.) Note that your implementation of these functions should not prompt the user for any information nor should they directly produce any screen output; they should just perform their prescribed actions as explained in the comments found in the file.
2. To test your implementation of the repeated squaring algorithm, the instructor will provide you with a paper copy of some problems to solve. These problems involve RSA key generation, which requires modular exponentiation.

### Implementation Constraints

* Follow the same coding standards used in the previous assignments