

**CSCI 124 - Discrete Structures II - Fall 2010**  
**George Washington University**

**Homework 2: 100 points**

due 22 October 2010, by 6 pm on Blackboard

**Policy on collaboration:** All examinations, papers, and other graded work products and assignments are to be completed in conformance with The George Washington University Code of Academic Integrity. You may discuss HWs among yourselves, and work on them in groups. However, each student is expected to write his or her own HW out independently; you may not copy one another's assignments, even in part. You may not collaborate with others on the quizzes, tests or final.

You are expected to cite all your sources in any written work that is not closed book: papers, books, web sites, discussions with others - faculty, friends, students. For example, if, in a group, one student has a major idea that leads to a solution to a HW problem, all other students in the group should cite this student.

**You may not copy code from anywhere; you may not use any code you have not written except for standard libraries.**

*Any violations will be treated as violations of the Code of Academic Integrity.*

This HW will be in electronic form and will be submitted electronically, on BLACKBOARD. It will consist of a single compressed file: .rar, or .zip consisting of the code itself, with a simple README file with documentation and any instructions on running the code.

**All code must run on hobbes. If it doesn't run on hobbes, you will get NO credit.**

Write code, in C++, C or Java, for:

(a) (20 points) The shift cipher: given input  $a$ ,  $b$  and  $m$ , output  $a + b \text{ rem } m$ . Your result should be non-negative and strictly smaller than  $m$ .

(b) (40 points) The generalized euclidean inverse: given input  $a$  and  $m$ , output  $a^{-1} \text{ mod } m$  and a text string "inverse exists" or return an error value and the text string "no inverse". Your result for  $a^{-1}$ , if it exists, should be non-negative and strictly smaller than  $m$ . You may not use code that you have not written, except code for quotients and remainders. In particular, you cannot use existing code for the euclidean algorithm that you have not written.

(c) (40 points) The affine cipher: given input  $a$ ,  $b$ ,  $m$ , and a text string saying "forward" or "inverse", output  $a \times b \text{ mod } m$  if the string says "forward", and  $a^{-1} \times b \text{ mod } m$  if it says "inverse". Use the code written in part (b) above. Make sure you check that  $a$  is invertible, and that both  $a$  and  $b$  are non-negative and smaller than  $m$ . Your output should be non-negative and strictly smaller than  $m$ .

It is easy to test your code. Run it for small values of  $a$ ,  $b$ , and  $m$ , and see if you get the expected results. That is how the TA will grade it, by testing it on small examples. Make sure you have a README file which describes how each program can be called, and also what it does. Make sure your code is well documented, there are points set aside for this.