Introduction
This assignment is essentially the same as problem 39 of chapter 3 in the textbook (page 255). A few additional details are provided in the problem statement given here, and the due date and submission procedure have been provided.

Problem Statement
You are to implement three programs that implement an error-detection mechanism using the standard CRC algorithm described in the text (see pages 212-215). The first program, named generator, reads from the standard input a line of ASCII text containing an n-bit message consisting of a string of 0s and 1s with optional spaces (blanks) at arbitrary positions. The second line is the k-bit polynomial, also in ASCII. The generator program writes two lines to the standard output. The first line is the n + k 0s and 1s representing the message to be transmitted. The second line of output is the polynomial, just as it was read (with the possible exception of optional spaces).

The second program, named verifier, reads the output of the generator program and displays a message indicating whether it is correct or not.

The final program, named alter, reads the output from the generator program (that is, two lines of 1s and 0s), and inverts one bit on the first line depending on its command-line argument, which is the bit number to be changed. The leftmost bit on the first line is numbered 1. All other bits in the input are copied, without change, to the standard output.

If you type

```
./generator < inputfile | ./verifier
```

you should receive a message indicating that the message is correct. But if you type

```
./generator < inputfile | ./alter 1 | ./verifier
```

you will get an appropriate error indication. These commands assume that inputfile is the name of a file containing the two appropriate input lines.

Details
The longest message and polynomial your program will need to process will contain 128 bits. Of course there can certainly be smaller messages and polynomials used in evaluating your work.

You may assume all the input used with your programs will be syntactically correct. That is, there will be no illegal characters in the input, or more than 128 1s and 0s in a message or generator polynomial. Of course, the message with the suffixed cyclical redundancy check can be longer than 128 bits, as indicated in the problem statement.

You may write your programs using C, C++, Java or Python 3. In all cases, a single file for each program, appropriately named, will be expected as your solution. If your solution requires something
other than a single command for compilation and/or execution (that is, cc, c++, javac/java, or python3), then you must provide appropriate details. As noted, a Java program (for example, generator.java) will be compiled using the command javac generator.java and executed using the command java generator < inputfile. A python3 program (for example generator.py) will be executed using the command python3 generator.py < inputfile. Remember that all programs will be compiled and/or executed on loki.

Your solution is to represent your own work. Do not work with others to develop your solutions.

Evaluation
Your program will be evaluated on loki.ist.unomaha.edu using a variety of input data (messages and polynomials). The instructor’s solutions to the generator, alter, and verifier programs are available in executable form in the directory /home/stanw/csci3550/prog1 on loki.ist.unomaha.edu. Nine sample input data files are also provided in that directory, book.in (from Fig. 3-9 on page 214 of the textbook), and case1.in through case8.in. Use the instructor’s solutions to determine the expected output. You are encouraged to also prepare additional test files of your own.

Requirements
You must write (and test) three programs as described above. The source code for these programs must be submitted by Tuesday, February 27, 2018 by 11:59 PM. To submit your solution, create a directory named csci3550-181-prog1 at the top level of your home directory on Loki and place the required files in it. That is, if your username is brenfro, then you must create a directory named /home/brenfro/csci3550-181-prog1 and place the required source files (for example generator.c, alter.c, and verifier.c) in it. There should be no other files in that directory, and the files you place there should not be changed or removed until you have received a grade report for the assignment.

**DO NOT CREATE THE SUBMISSION DIRECTORY UNTIL YOU ARE READY TO SUBMIT YOUR WORK.**

As always, please contact the instructor if you have questions, and periodically check the class web site for any additions or corrections to this assignment.