The goal of the first assignment is to write two simple C functions, and make sure that your code compiles correctly, and figures out the correct answers. To keep things simple, you have been provided with the infrastructure C code. Feel free to read through the infrastructure code, and see how some of the features of C we will learn about in the future work in a program, but you are not required to understand all the details of the infrastructure for this assignment.

The infrastructure code uses the command line argument specified when the C program is invoked to determine an input and output file name. The input file name consists of the argument with ".txt" appended to the end. The output file name consists of the argument with "_results.txt" appended. The infrastructure reads the input files and looks for lines in the file that contain either the word "fib" followed by a single number, or the word "diffSq", followed by two numbers. You are responsible for implementing two C functions; one called "fib" and the other called "diffSq". When the infrastructure finds a "fib" line in the input file, it will invoke your fib function, passing in the number from the input file as an argument, and print a line to the output file with the name of the function and the results. When the infrastructure finds a "diffSq" line in the input file, it invokes your diffSq function with the two numbers found in the input line, and print a line to the output file with your results.

All the files you need to do this assignment are contained in the tar file, hw01.tar.gz. Download this file from the class web page into your CS220 directory, and then (on a UNIX machine) type the command:

```
tar -xvzf hw01.tar.gz
```

This command will create a sub-directory of the current directory called "hw01", which has several files, as follows:

- hw01.c – This file has the infrastructure code in it, and a place for you to write your fib and diffSq functions.
- Makefile – This file has make rules to compile, test, debug, and package your results for submission. The make targets available are:
  - test – Compile your program if it is out of date, and run it using input1
  - debug – Compile your program if it is out of date, and start the gdb debugger on the program.
  - hw01 – Compile your program if it is out of date
  - clean – Remove the executable file created by the hw01 target and any results files created by the test target
  - submit – Create a tar file for submission on myCourses. (If you are not running on a machine where your userid is your bmail userid, you must invoke this as "make USER=jsmith23 submit", where "jsmith23" is your bmail userid.)
- input1.txt – An example input file
- input1_correct_results.txt – The correct results that should be produced from input1.txt

Your "fib" function should calculate the n^{th} Fibonacci number, where *n* is the argument to the function. Fibonacci numbers are a sequence, 0, 1, 1, 2, 3, 5, 8, … In this sequence, by definition, the zeroth number is 0, and the first number is 1. All subsequent numbers in the sequence are just the sum of the previous two numbers in the sequence. For more information, see the Wikipedia Fibonacci Number Article.

Your "diffSq" function should calculate and return the difference of the squares of the two arguments, *a* and *b*. Mathematically, the results should be $a^2 - b^2$. Warning: There is no exponentiation operator in C.

You can find the starting place to write your two functions at the bottom of the hw01.c file. Once you have coded your results, run "make test", and make sure that the results in "input1_results.txt" match the correct results in "input1_correct_results.txt". If not, you did something wrong and should fix it.

Create your own test cases to make sure your functions work correctly using different input.

When you are done testing, run "make submit" to create a file called "hw01_<*userid*>.tar.gz", where <*userid*> is your gmail userid. Upload this file on myCourses under Content, Homework Submissions, Homework 01 Submission. This assignment is due at 11:59 PM on Sunday, January 29, 2017. You may submit as many times as you wish; but only the latest submission will be graded.

This assignment is worth 10 points. Your grade will be calculated as follows:

- If you submit late without an extension, there will be a two-point deduction for every 24 hours you are late. Extensions are available only in special circumstances, and can only be given by the instructor or a TA.
- There will be a three-point deduction for submissions that do not follow the required format. For instance, if you do not run make submit to create the correct tar file, or if you run make submit on a machine where your userid is incorrect so that the resulting tar file has the wrong name OR wrong contents (your userid is used to create a sub directory that is contained in the tar file).
- There will be a seven-point deduction if there are compiler errors when compiling your code.
- There will be a five-point deduction if your code does not run to completion on any of the test cases used to grade your code. For instance, if your code causes a segmentation violation, there will be a five-point deduction.
- There will be a two-point deduction for each class of compiler warning message issued when compiling your code.
- There will be a three-point deduction for each test case which does not produce the correct results. You code will be run with the test case in "input1.txt", as well as two other input cases, which will not be pre-published. The unpublished test cases will invoke both your fib and diffSq functions; and will provide only valid arguments. All arguments will be positive integers.