# CSCI 1320 (Principles of Algorithm Design I), Fall 2007
# Homework 7

**Assigned:** November 27, 2007.

**Due:** December 4, 2007, at 5pm. *Not accepted past 5pm December 6.*

**Credit:** 40 points.

## 1 Reading

Be sure you have read (or at least skimmed) chapters 9, 10, and 11.

## 2 Programming Problems

Do the following programming problems. You will end up with at least one code file per problem. Submit your program source (and any other needed files) by sending mail to `bmassing@cs.trinity.edu`, with each file as an attachment. Please use a subject line that mentions the course number and the assignment (e.g., "csci 1320 homework 7"). You can develop your programs on any system that provides the needed functionality, but I will test them on one of the department's Linux machines, so you should probably make sure they work in that environment before turning them in. Remember also that I will usually deduct points if your programs do not compile without warnings using the flags `-Wall` and `-pedantic`.

1. (10 points) Your mission for this problem is to fill in missing parts of the following C program, which converts a Julian date (day of year) into a month/day date: julian-date.c[1].

   Places where you need to fill something in are identified with comments `YOUR CODE GOES HERE`. (You should only need to add one line to the main program, and fill in the body of the function `convert_jdate`.) Sample output:

   ```
   Julian date 1 is Jan 1
   Julian date 32 is Feb 1
   Julian date 33 is Feb 2
   Julian date 365 is Dec 31
   ```

2. (30 points) Your mission for this problem is to write a C program that performs some of the same functions as the Unix command `grep` — searches in a file for all lines that match a given search string, and prints the ones that do. For example, if you have a file containing the following lines:

   ```
   Hello world!
   There is a German word for worldview that I have forgotten.
   World without end.
   Goodbye!
   ```

[1] `http://www.cs.trinity.edu/~bmassing/Classes/CS1320_2007fall/Homeworks/HW07/Problems/julian-date.c`

and you search for `world`, the first and second lines should print. (It's easiest to do the match in a case-sensitive way, so the third line does not match.)

The command should prompt first for the name of the file to search, then for the string to search for. Be as careful as you can about reading text — it is very easy to read more characters than will fit into the character array you're putting them in. If the input file contains lines that are too long, you can just print warning messages about them. You may find the library function `strstr` useful.