CS142 **Assignment 1** Spring 2025

Given: Fri, Jan 17
Due: Thu, Jan 23, 3:00 p.m.

Before starting the assignment, please read the assignment policy available on the main page of the course website. It includes, among other things, rules on external help and collaboration, as well as information on deadlines and precise instructions for submitting your assignments.

There are also some comments at the end of this assignment that you probably want to read before you start working on the assignment.

1. (100%) Create a class for storing songs. The class should be called `Song` and each object in this class should represent one song. (This class could be used, for example, in a music library.) Each song has a title, an artist and a recording year. The title and artist are strings and both can consist of multiple words. The year is an integer.

   The class should be designed using the principle of encapsulation. In other words, make sure the data is private.

   Create the following operations for working with objects of this class:

   (a) A function `print(s, out)` that prints song `s` to output stream `out` in the following format:

   ```
   Title
   Artist
   Year
   ```

Three complete lines are printed. The stream argument should have `cout` as a default value.

(b) A function `read(s, in)` that reads song `s` from input stream `in`. The data in the stream is assumed to be in the format specified for `print`. In particular, three complete lines are read. No error-checking is performed. The stream argument should have `cin` as a default value.

(c) A function `equals(s1, s2)` that returns **`true`** if songs `s1` and `s2` have the same title, artist and recording year. (The return value should be a Boolean value, not a string.)

Note that these functions should be standalone functions, not methods or member functions, if you know what those are.

The above describes the arguments of the various functions but not how these arguments are passed to the functions. Part of what you have to do in this assignment is decide how these arguments should be passed. Make sure to think about it carefully taking into account the discussion we had in class. Consult the notes.

Write (and submit) a test driver for your class and its operations. (A test driver is a program whose only purpose is to run tests.) Make sure you test all three functions.

Here are some comments you might find useful for this assignment and the following ones:

1. If you haven't used strings in C++ before, you'll need to include the `string` library when you test your functions (in addition to `iostream`).

2. In the implementation of the `read` function, you'll want to use the `getline` function to read the title and artist of the song. In case you're not familiar with it, `getline(in, s)` reads characters from input stream `in` into string `s`. Characters are read until then end of the current line. The newline character is read but not included in `s`.

3. In general, functions should do all that they're supposed to do *and nothing else*. That's because some extras, even if well-meaning, might not make sense in every context in which the function could be used. For example, the `read` function should not prompt the user by printing something like this:

   ```
   Please enter the title of the song:
   ```

   If the function is used to read from `cin`, that might be OK. But if the function is used to read from a file, that would a bad idea.

4. To test the function `equals`, you'll probably want to read two songs and compare them. For example, when the program is ready to read the songs, you might type the following:

   ```
   Moment's Notice\n
   John Coltrane\n
   1958\n
   Open Eye Signal\n
   Jon Hopkins\n
   2013\n
   ```

   Notice the newline characters at the end of each line: those are generated when you press the *enter* or *return* keys.

Now, if your `read` function reads the year of the first song by using the input operator (>>), the only thing that will be read will be the year itself (the number), not the newline character that follows the year. And this will cause problems when the program attempts to read the title of the second song. (Why?)

The solution is to make sure that the `read` function reads an entire line when it reads the year. You can't use `getline` to read the year because it's a number and the `getline` function only works for strings. But you can achieve the same result by getting rid of the newline character that follows the year. And you can do that the same way we got rid of the colon when we were reading times in class.