MCIS-4130
Homework Assignment 7

Purpose:

This problem provides an opportunity to be a client of container (and at your option, template) classes.

What to hand in:

Hand in a listing for your test program; also, hand in listings for your header files, and all implementation files. Include output of your test program with the provided data.

Problems:

A. Containers of Date things.

Using the Birthday implementation we built last week, and a container class (such as we built this week), create a container for the Birthday type. This may be either a template-based container, or a specialized Birthday container.

Read an arbitrarily large set of names (either from standard input or a file–your design choice) and store them in the container.

Print the contents of the containter to the output stream.

Search for all records with a given name (entered by the user running the program), and print any matches to the output stream. Create a function for this purpose.

Search for all records with a given birthdate (entered by the user running the program), and print any matches to the output stream. Create a function for this purpose

Notes:

The code for the various classes in the notes can be found on hosts ***odin.cair.du.edu***, in `~mschwart/C++/Examples`. You may use this, or type the examples in the notes in yourself.

Consider what *type* the different functions should return, as well as the parameters each function should take.

The distinction about where template instantiations are declared and where they are defined (and thus code is produced) is very compiler dependent at this time. Some compilers provide special switches (e.g., odin's cxx); others provide the ANSI C++ mechanism (e.g., mercury's g++); yet others work it out themselves (e.g., Centerline). This area may lead to problems; be sure to ask about your situation so you know what to do!

Following is a sample test program that achieves the purpose of the exercise; to use it, you must create the "`bdaylist.h`" file, the (global) `Birthday_List` container (in this example an array-style container), and the functions `PrintMatchingName()` and `PrintMatchingBirthday()`.

```
/*
** MCIS4130
** Michael Schwartz
** Sample test program for Birthday lists
*/
#include "duconfig.h"
#ifdef USING_STANDARD
  #include <cstdlib> /* For atoi() */
  #include <cstring> /* For strtok() */
  #include <fstream>
  #include "bdaylist.h"
  using namespace std;
#else
  #include <stdlib.h> /* For atoi() */
  #include <string.h> /* For strtok() */
  #include <fstream.h>
  #include "bdaylist.h"
#endif

void parse(char *buffer)
{
  char *name_part;
  char *date_part;
  int month, day, year;
  name_part = strtok(buffer,"\t;");
  date_part = strtok(NULL, "\t;");
  if (name_part == 0 || date_part == 0)
    return;
```

```
  month = atoi( strtok(date_part, "/ ") );
  day   = atoi( strtok(NULL,      "/ ") );
  year  = atoi( strtok(NULL,      "/ ") );
  Birthday b (name_part, month, day, year);
  // Push a new element to end of the list
  Birthday_List.push_back(b);
}

int main()
{
  char buffer[255+1];
  ifstream ifs("birthday.dat");
  int month, day, year;

  while (ifs)
  {
    ifs.getline(buffer, sizeof(buffer));
    parse(buffer);
  }

  // Print contents to the output stream
  cout << Birthday_List;

  cout << "Enter a name: ";
  cin.getline(buffer, sizeof(buffer));
  int num = PrintMatchingName(buffer, cout);
  cout << num << " matches" << endl;

  cout << "Enter a day (M/D): ";
  cin.getline(buffer, sizeof(buffer));
  month = atoi( strtok(buffer, "/ ") );
  day   = atoi( strtok(NULL,   "/ ") );
  year  = 1995;
  Date d (month, day, year);
  num = PrintMatchingBirthday(d, cout);
  cout << num << " matches" << endl;

  return 0;
}
```

Use the following data (the sample program assumes a tab or colon-delimited file):

George Washington	2/22/1732	John Adams	10/30/1735
Thomas Jefferson	4/13/1743	James Madison	3/16/1751
James Monroe	4/28/1758	John Adams	7/11/1767
Andrew Jackson	3/15/1767	Martin VanBuren	12/5/1782
William Harrison	2/9/1773	John Tyler	3/29/1790
James Polk	11/2/1795	Zachary Taylor	11/24/1784
Millard Fillmore	1/7/1800	Franklin Pierce	11/23/1804
James Buchanan	4/23/1791	Abraham Lincoln	2/12/1809
Andrew Johnson	12/29/1808	Ulysses Grant	4/27/1822
Rutherford Hayes	10/4/1822	James Garfield	11/19/1831
Chester Arthur	10/5/1830	Grover Cleveland	3/18/1837
Benjamin Harrison	8/20/1833	William McKinley	1/29/1843
Theodore Roosevelt	10/27/1858	William Taft	9/15/1857
Woodrow Wilson	12/28/1856	Warren Harding	11/2/1865
Calvin Coolidge	7/4/1872	Herbert Hoover	8/10/1874
Franklin Roosevelt	1/30/1882	Harry Truman	5/8/1884
Dwight Eisenhower	10/14/1890	John Kennedy	5/29/1917
Lyndon Johnson	8/27/1908	Richard Nixon	1/9/1913
Gerald Ford	7/14/1913	James Carter	10/1/1924
Ronald Reagan	2/6/1911	George Bush	6/12/1924
William Clinton	8/19/1946	George Bush	7/6/1946

Use John Adams for the user-selected name in your test program run

Use 11/2 for the user-selected date in your test program run

Note that the birthday equality match should use only day and month, not day, month, and year.

Evaluation criteria:

40	New header file with declarations for the two functions and global instance
40	New source file defining the two functions and global instance
20	Main program and output ***Note: If your program doesn't compile, hand in the error listing as the results!***