CIS 459.22 LAB2

****

Assigned: 4/14/05

Due: 4/27/05 11:59pm

****

Goal

- To write an OOP program with C++

Points

This lab is worth 100 points.

Description

- **Your code will implement similar function as lab1 with OOP of C++. You can reuse some fragment of code from lab1.**

- **Your *main()* program should give the user a menu to choose from, as shown below.**

    I. Enter a date
    Q. Quit

    Your program should proceed based on the user's choice.

    a) If the user chooses I (Input),

        read a date from the keyboard as mm/dd/yy;
        convert it to the output format if input is right, otherwise, give error message;
        return to the main menu and wait for user to choose the next action;

    b) If the user chooses Q(quit), exit from the program.

- **Write a class Date for saving a date such as 10/02/01. You should have the following class interface:**

```
class Date
{
   private:
        unsigned day;
        unsigned month;
        unsigned year;

        //display an error message if the date format given by the user is
        invalid
        void errmsg(const char* msg);

   public:
        //set this object to the specified date
        void set(const char* mmddyy);

        //print this date to your standard output, i.e. monitor
        void display(void) const;

};
```

NOTE: **Please don't change this class interface.**

- **The member function *errmsg* should display the following error message if, for instance, 1/1/01 has been entered as a date by the user.**

```
Invalid date format: 1/1/01
```

- **The member function *set* should extract the month, day and year.**

```
void Date::set(const char* mm_dd_yy)
{

    // Examine if the user's input format is mm/dd/yy
    // If not, call errmsg(mm_dd_yy);

    // HINT: use strtok and atoi to parse the input and extract into month,
    day and year.

    // Make a Y2K correction for a 2-digit year assuming we are interested
    only in years between 1950-2049, i.e. if (year < 50) year += 2000; else
    if (year < 100) year += 1900;
```

// Check the validity of date. Make sure *month*, *day* and *year* are in their appropriate range, e.g. month: 1-12. For example, 02/31/03 and 13/28/04 are invalid.

// Pay more attention about the leap years. That is, if *year*%4==0, then it is a leap year. However, if *year* is a century year, *year*%100==0, then *year* is not a leap year unless *year*%400==0.

}

- **The member function *display* should produce the output:**

  The date is 1/1/2001

  where the format for the date is mm/dd/yyyy, and *month* and *day* do not need to be zero filled (as shown above).

- **Save your program as lab2.cpp and compile it with *g*++ by entering the following command in the xterm window:**

  ***g*++ -o lab2 lab2.cpp <return>**

  Remove all compilation errors and warning messages before running the program. Recompile each time you make a change to the program. By finishing, your program should compile and return to the prompt without displaying any error messages.

- **Run your program by entering the following command in the xterm window:**

  lab2 <return>

  Test your program with several cases you may think of, such as "1/1/01", "01012001", "01/01/50", "12/12/01". It should comply with the specifications given above.

  One possible script for running your program is as follows :

  (I) Enter a date
  (Q) Quit

  I <return>

  Enter the starting date <mm/dd/yy> => 1/1/01 <return>
  Invalid date format: 1/1/01

(I) Enter a date
(Q) Quit

I <return>

Enter the starting date <mm/dd/yy> => 01012001 <return>
Invalid date format: 01012001

(I) Enter a date
(Q) Quit

I <return>

Enter the starting date <mm/dd/yy> => 01/01/50 <return>
The date is 1/1/1950

(I) Enter a date
(Q) Quit

I <return>

Enter the starting date <mm/dd/yy> => 12/12/01 <return>
The date is 12/12/2001

(I) Enter a date
(Q) Quit

Q <return>

Submission

- **When you finish with the lab, you need to turn it in for grading. The *submit* command submits your lab electronically. You MUST use the *submit* command to turn in your labs. The format of *submit* command is as follows:**

  ***submit* classname labname files-to-submit**

  where,

  classname is the name of the CIS 459.22 section that you are enrolled in. Your classname is c459.22ab .

  labname is the lab you are working on (lab1, lab2, etc.). For this lab, labname is lab2 .

  files-to-submit is a list of the files that make up the lab. For now, it only contains lab2.cpp .

  NOTE:

1. **All of the files in a lab MUST be submitted using one command. If you use two *submit* commands, the second one erases the files from the first submission.**
2. **Your programs MUST be submitted in source code form. Make sure that you submit the "lab2.cpp" file only. Do not submit the object file. If you submit the object code, your lab submission will be considered as invalid and late penalty will be imposed.**
3. **Each *submit* command MUST be entered on one line without pressing Enter. If the line you are entering is too long, it wraps onto the next line.**

Submitting Lab2

To submit your lab for grading, use the following command from the directory which contains your lab2.cpp file:

***submit* c459.22ab lab2 lab2.cpp**