CIS 459.22 LAB3

****

Assigned: 4/28/05

Due: 5/11/05 11:59pm

****

Goal

- To write constructors and destructor for a class type
- To understand the concept of inheritance

Points

This lab is worth 100 points.

Description

- **Your code will extend the functionality of lab2 with a menu of 3 choices. You can reuse some fragment of code from lab2.**

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

  (I) Enter a date
  (T) Enter a time
  (Q) Quit

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

  a) If the user chooses I (Input a date),

  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 T (Input a time),

read a time from the keyboard as <u>hh:mm:ss mm/dd/yy</u>, note that there is a blank space between hh:mm:ss and 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;

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

- **Your *main()* program should declare two objects of class Date: d1 using default constructor and d2 using copy constructor, as shown below. Do the same for class Time.**

```
Date d1;
Date d2(d1);

Time t1;
Time t2(t1);
```

- **The interface of class Date is given as follows.**

```
class Date
{
  protected:
      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);

      //set this object to the specified date
      <return type> set(const char* mmddyy);

  public:
      Date() ;  // default constructor should set the Date object to
      1/1/1950
      Date(const Date &d);  // copy constructor should also print "The
      Date copy constructor is called now"
      Date(const char *mm_dd_yy);  // constructor that initializes with
      the characters input
      //Destructor should print "The Date destructor is called now"

};
```

NOTE: **Please pay attention to the members written in bolds and the access specifiers, which identify the changes/extensions from the Date class in lab2. In the following description, they will be elaborated in more details.**

1. **Change the private access of data members to protected**

2. **Make *set* function protected. The *set* function should check for the validity of a date (between 1950 and 2049 only), and take care of leap years also**

```
<return type> Date::set(const char* mmddyy)
{

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

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

    // 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
    information 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.

}
```

3. **Add a default constructor Date(), which should initialize an object to be 1/1/1950**

4. **Add a copy constructor Date(const Date &d), which, in addition to its main functionality, should print out "The Date copy constructor is called now"**

5. **Add a constructor Date(const char *mm_dd_yy), which initializes with the characters input**

6. **Similarly, add a destructor that should print out "The Date destructor is called now", in addition to its main functionality**

- **Create a new class called Time, which inherits class Date. A Time object will contain information about both date and time, i.e month, day, year, hours, minutes and seconds.**

```
class Time : public Date
{
   protected:
        int hours;
        int minutes;
        int seconds;

        //set this object to the specified time according to the input string
        <return type> set(const char* inputTime);

   public:
        Time();  // default constructor should set the Time object to
        00:00:00 1/1/1950
        Time(const Time &t);  // copy constructor should also print "The
        Time copy constructor is called now."
        Time(const char *inputTime);   // constructor that initializes with
        the characters input
        // destructor also should print "The Time destructor is called now."

        void display() const;  // display the date and the time

};
```

In the following description, more details will be given about this class.

1. **The *set* function should check the format of an input, also the validity of it**

```
<return type> Time::set(const char* inputTime)
{

    // Examine if the user's input format is hh:mm:ss mm/dd/yy, note that
    there is a blank space between hh:mm:ss and mm/dd/yy.
    // If yes, set the corresponding data members; otherwise, give an error
    message;

    // HINT: use strtok and atoi to parse the input and extract values for
    data members

    // In addition to check the validity of date, check the validity of time.
    Make sure hour, minute and second information are in their
    appropriate range, e.g. hour: 0-23. For example, 00:58:30 10/18/04 is
    valid, while 22:80:20 01/02/99 is invalid.
```

}

2. **Add a default constructor Time, which should initialize an object to be 00:00:00 1/1/1950**

3. **Add a copy constructor Time(const Time &d), which, in addition to its main functionality, should print out "The Time copy constructor is called now"**

4. **Add a constructor Time(const char *inputTime), which initializes with the characters input**

5. **Similarly, add a destructor that should print out "The Time destructor is called now", in addition to its main functionality**

6. **The member function *display* should produce the output:**

   The time is 01:58:30 1/18/04.

   where
   - **the format for the time is hh:mm:ss, with 2 digits each for hour, minute and second**
   - **the format for date is mm/dd/yyyy, and *month* and *day* do not need to be zero filled (as shown above).**

7. **You can define any new functions if necessary, as long as the above requirements are fulfilled**

- **Save your program in a modular fashion.**
  - class Date: date.cpp for class implementation of class Date; date.h for class interface definition
  - class Time: time.cpp for class implementation of class Time; time.h for class interface definition
  - lab3.cpp for main function

- **Compile it with *g*++ by entering the following command in the xterm window:**

  ***g*++ -o lab3 lab3.cpp time.cpp date.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:**

  lab3 <return>

  Test your program with several cases you may think of, such as "09:30:45 1/1/01", "3:80:100 01012001", "17:30:49 01/01/50", "10:30:00 12/12/01". It should comply with the specifications given above.

  One possible script for running your program is as follows :

  The Date copy constructor is called now.
  The Time copy constructor is called now.

  (I) Enter a date
  (T) Enter a time
  (Q) Quit

  I <return>

  Enter the date <mm/dd/yy> => 1/1/01 <return>
  Invalid date format: 1/1/01
  The Date destructor is called now.

  (I) Enter a date
  (T) Enter a time
  (Q) Quit

  T <return>

  Enter the time <hh:mm:ss mm/dd/yy> => 25493310/25/2004 <return>
  Invalid time format: 25493310/25/2004
  The Time destructor is called now.
  The Date destructor is called now.

  (I) Enter a date
  (T) Enter a time
  (Q) Quit

  I <return>

  Enter the date <mm/dd/yy> => 01/01/50 <return>
  The date is 1/1/1950
  The Date destructor is called now.

  (I) Enter a date

(T) Enter a time
(Q) Quit

T <return>

Enter the time <hh:mm:ss mm/dd/yy> => 12:09:59 12/12/01 <return>
The time is 12:09:59 12/12/2001
The Time destructor is called now.
The Date destructor is called now.

(I) Enter a date
(T) Enter a time
(Q) Quit

Q <return>

The Time destructor is called now.
The Date destructor is called now.
The Time destructor is called now.
The Date destructor is called now.
The Date destructor is called now.
The Date destructor is called now.

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 lab3 .

  files-to-submit is a list of files that make up the lab. For now, it contains lab3.cpp, time.cpp, time.h, date.cpp, and date.h.

  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 source files 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 Lab3

To submit your lab for grading, use the following command from the directory which contains all files for this lab:

***submit* c459.22ab lab3 lab3.cpp time.cpp time.h date.cpp date.h**