Creating a Date Class

Date class

- We want to be able to create a class that will allow us to set, access and modify a date
 - There may be other useful functions
- But first we must understand how to access and use the system time

Using System Time

To get the current time you need to:

#include <ctime>

This header file contains definitions of functions & related datatypes to get and manipulate date and time information.

To get the system time we need to:

- 1. Declare two variables
 - One of type time_t to store the time in its natural form (seconds)
 - One of type tm to convert it to days, months, years, hours, minutes & seconds
- 2. Use the time() function to retrieve the time.
- 3. Use the localtime() function to convert it to type tm

Using System Time – Data

Types
There are two primary data types we need to use

- time_t is a datatype capable of representing a time in terms of seconds since Jan 1, 1970
 - For example you could declare a time_t variable like this time_t now;
- tm is struct that stores time fields in separate members

tm_year: years since 1900

tm_mon: months since January (0-11)

tm_mday: day of month (1-31)

■ For example - you could declare a tm variable like this

tm *currentTime; This must be a pointer

```
Using System Time – Functions
We need two time functions to retrieve and convert the time
   time_t time (time_t *timer);
       ■ Returns a value of type time_t as the current system time.
       ■ The system time is represented as the number of seconds since 00:00, Jan 1,
       To use it we just call it (using NULL as the argument)
       For Example:
                                     Declare a variable to store the time
           time t now;
                                     Retrieves the time
           now = time(NULL);
   tm *localtime (const time_t *timer);
       Converts time_t from seconds to the struct tm separating into
         month, day, year, etc - so we can access those values independently
          For Example:
                                   Declare a variable to store the converted time
             tm *currentTime;
             currentTime = localtime(&now); Converts the time to TM
```

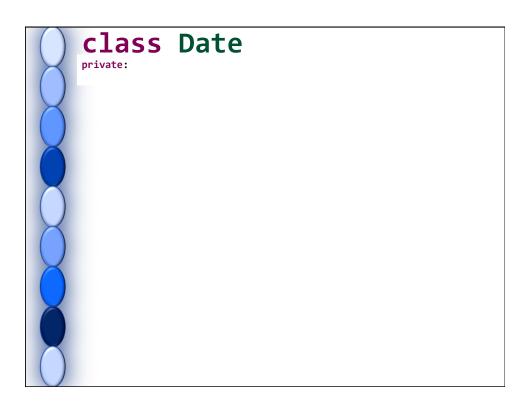
```
Using System Time – Example
#include <ctime>
                          Declare a variable to store the time
time_t now;
                          Declare a variable to store the converted time
tm *currentTime;
                          NOTE: This must be a pointer varia
                          Retrieves the time
now = time(NULL);
                                   Converts from seconds to type tm
currentTime = localtime(&now);
                                    - Separates to months, days, years
// Store time members into variables
int currentYear;
int currentMonth;
int currentDay;
currentYear = 1900 + currentTime->tm year;
                                                  Accesses the year
currentMonth = currentTime->tm_mon + 1;
                                                  Accesses the day
currentDay
              = currentTime->tm_mday;
```

Back to our Date Class

We want to be able to create a class that will allow us to utilize a date

- Which attributes are needed in a Date?
- Which methods?
 - Accessors?
 - Mutators?
- What special calendar circumstances should we account for?

class Date public:



Calculating the Leap Year

- It is a little-known fact that not every fourth year is a leap year.
- Since the Earth orbits the Sun in 365.2425 days, not 365.25 days, additional leap year rules exist to correct for these additional decimal places.
- In short, a year is a leap year if it is divisible by four, UNLESS it is also divisible by 100. A year is NOT a leap year if it is divisible by 100 UNLESS it is also divisible by 400.

Condition	Result	Examples
Not divisible by 4	Not a leap year	2009, 2010, 2011
Divisible by 4	Leap year	2008, 2012, 2016
Divisible by 100	Not a leap year	1800, 1900, 2100
Divisible by 400	Leap year	2000, 2400

You will need to come up with the algorithm to calculate this based on this data