Practice with structs and strings

Create a `struct` to hold a date:

```
struct date {
	int month;   // 1-12
	int day;     // 1-31
	int year;    // anything
};
```

Use this `struct` to write the following functions. Make sure to test each function in your `main` function before moving on. I've given you the function definition line for each function.

1. Write a function that prints a date nicely formatted (e.g., with slashes separating the components).

   ```
   void print_date(const date & d)
   ```

2. Write a function that constructs a new date from its three arguments and returns the new date.

   ```
   date make_date(int month, int day, int year)
   ```

3. Write a function that constructs a new date from a single string argument that looks like "`mm/dd/yyyy`" where mm is a 2-digit month, dd is a 2-digit day, and yyyy is a 4-digit year.

   ```
   date make_date(const string & datestring)
   ```

   Notice that C++ will permit you to have two or more functions with the same name, as long as the number, data types, or order of the arguments are different in each version. This is called ***function overloading****. When your code calls a function that has been overloaded, C++ figures out which one you mean by examining the number and data types of the arguments that you are passing to the function.*

4. Write a function that constructs a new date by picking each of the three components randomly. Ensure that you only construct valid dates, i.e., you should never construct a date of April 31. Don't worry about leap years.

   ```
   date make_random_date()
   ```

5. Write a function that determines if one date is earlier than another date.

   ```
   bool earlier_than(const date & d1, const date & d2)
   ```

6. Write a function that finds the earliest date in a vector of dates. Hint: use your `earlier_than` function. Test this function by generating a bunch of random dates, putting them in a vector, and finding the earliest (you should print the dates as well to make sure you really find the earliest one).

   ```
   date earliest_date(const vector<date> & dates)
   ```

7. Write a function that returns a date one day in the future from a given date.

   ```
   date add_day(const date & d)
   ```

8. Write a function that figures out the day of the week for any date and returns it as a string. Use Google to find an algorithm for this.

   ```
   string day_of_week(const date & d)
   ```

9. Thanksgiving (in the US) is always celebrated on the fourth Thursday of November. Write a function to calculate the date of Thanksgiving for any year.

   ```
   date find_thanksgiving(int year)
   ```