Practical 15: Date Formatting

The main focus of this practical is methods but it will also test your skills in many other fundamentals such as arrays, strings, selection and iteration.

Task 1

Recall from a previous practical that the `Date` class can be used to interrogate the system's timer in order to get the current date and time. For example, the following `main()` method:

```
public static void main(String[] args)
{
     Date timestamp = new Date();
     String timestampAsString = timestamp.toString();

     System.out.println(timestampAsString);
}
```

will produce an output similar to:

```
Mon Nov 19 19:38:27 GMT 2012
```

Verify the above.

Don't forget to import the `java.util.Date` class at the top of your program.

Task 2

You need to implement the following methods whose declaration is given to you below:

```
static String getDayString(String dateString)
static String getMonthString(String dateString)
static int getDayInt(String dateString)
static int[] getTimeComponents(String dateString)
static int getYear(String dateString)
```

You are to assume that all methods above receive as their only parameter the string produced by the sample program in task 1 above. Each method then processes the string to extract some date/time related information from it.

`getDayString()` returns the long name of day of the week (`Monday`, `Tuesday`, `Wednesday`, ... etc.) Likewise, `getMonthString()` returns the long name of the month (`January`, `February`, `March`, ... etc.), `getDayInt()` returns the day of the month as a number, `getTime()` returns the hour, minute and second of the time as an array of three elements and finally, `getYear()` returns the year.

For the example in task 1, the above methods should return:

Method	Return value
`getDayString()`	`Monday`
`getMonthString()`	`November`
`getDayInt()`	`19`
`getTimeComponents()`	`19, 38, 27` (as elements of an `int` array of size 3)
`getYear()`	`2012`

Task 3

Create another method that also takes the same string obtained from the Date class and returns a formal date string such as:

```
Monday, November 19th, 2012
```

following on from the example in task 1.

This new method's declarations should be:

```
static String dateFormalString(String dateString)
```

As you would probably guess `dateFormalString` ought to employ the services of some of the methods you have created in task 2 if you wish to be efficient.

Task 4

You can use the `Date` class to produce any date information, not just the current date/time. When you create a `Date` object from the `Date` class, you can pass a `long` to the *constructor* of the class like for example:

```
Date timestamp = new Date(1000000000000L);
```

which will map to `Sun Sep 09 02:46:40 BST 2001`.

The number that you can pass to the *constructor* of the `Date` class represents the number of milliseconds since January 1, 1970, 00:00:00 GMT.

Using this information, can you write a program that will test the functionality of your methods above?