

ENGINEERING 1D04, Lab 4: Conditional Statements and Methods

Cheap Chocolate

Chocolate is a wonderful thing. As a student, the price point of a luxury item, such as chocolate, is of the utmost importance. Traditionally, the Monday after Easter is the best time to purchase chocolate. (It is not the purpose of this assignment to debate Easter versus Halloween for buying cheap chocolate.)

Requirements

1. Write a C# program that takes an input year and outputs the premium chocolate purchasing date of that year. See Figure 1 below.
2. With reference to Figure 1. a) is a textbox that shall accept an input year in the range 1700-2299. If the user inputs a year outside that range a message box shall pop up informing the user that the input year is out of range. (See *message box* in appendix.) b) If the user clicks the Go button, the following two actions should occur: c) this textbox shall contain the date on which Easter falls for the given input year (see algorithm given below); d) this textbox shall contain the date of the Monday immediately following Easter for the given input year. e) are the required labels on the form.

Figure 1 shows a Windows form titled "Form1". It contains three textboxes and a "Go" button. The first textbox is labeled "Input Year:" and contains the text "2007". The second textbox is labeled "Easter is" and contains the text "April 8". The third textbox is labeled "The Best day to buy Chocolate" and contains the text "April 9". A "Go" button is located at the bottom of the form. Red arrows point from labels 'a' through 'e' to the corresponding UI elements: 'a' points to the "Input Year" textbox, 'b' points to the "Go" button, 'c' points to the "Easter is" textbox, 'd' points to the "The Best day to buy Chocolate" textbox, and 'e' points to the labels "Input Year:", "Easter is", and "The Best day to buy Chocolate".

Figure 1. Input form

3. Dates must be formatted as "Month"+" "+"day", where Month is either March or April, and day is a number representing the day of the month.
4. You do not need to cope with input errors such as an empty textbox or a textbox that does not contain a valid integer.

Aside: more information about *if statements*: In case you would like to include multiple conditions in an *if* or *else if* clause, you can do it like this:

To test "condition1 and condition2", use `if (condition1 && condition2)`
To test "condition1 or condition2", use `if (condition1 || condition2)`

Algorithm (due to Gauss)

Let Y be the year between 1700 and 2299 inclusive.

Let the values of M and N be decided from Y using the following table:

Y	M	N
1700-1799	22	3
1800-1899	23	4
1900-2099	24	5
2100-2199	24	6
2200-2299	25	0

Let $a = Y \bmod 19$.

Let $b = Y \bmod 4$.

Let $c = Y \bmod 7$.

Let $d = (19a + M) \bmod 30$.

Let $e = (2b + 4c + 6d + N) \bmod 7$.

Easter is on March 22nd plus $(d + e)$ days. For example, if $(d + e)$ is 7 then Easter is on March 29th. March has 31 days. (In C#, the modulus operator (mod) is represented by %)

The algorithm has two exceptions.

- First, Easter is never on the 26th of April. In case the algorithm gives this as the answer, the actual answer is the 19th of April.
- Second, if the algorithm gives the date of Easter as April 25th with $d = 28$, $e = 6$ and $a > 10$ then Easter is actually on April 18th.

Please use two methods to implement the Gauss Easter Algorithm. One method should have the following interface:

```
void GaussEasterAlgorithm(int y, out int month, out int day)
```

where y is the year, $month$ is an integer that will either be 3 or 4 (3 for March, 4 for April) and day is an integer representing the day of the month.

The other method should be called by GaussEasterAlgorithm to implement the table shown above. This method should have the following interface:

```
void GaussTable(int y, out int m, out int n)
```

where y , m and n correspond with the columns of the above table.

Message Boxes

Message Boxes can be invoked by a simple call to the method `MessageBox.Show(string s)`.

For example, if you want to display a message box that says “You did something stupid”, with an “OK” button, then simply call

```
MessageBox.Show("You did something stupid");
```

in the appropriate place.

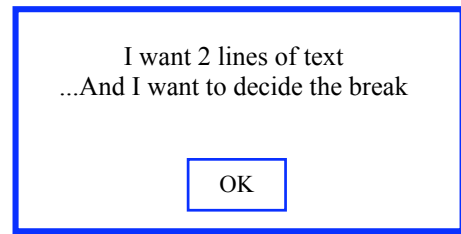
Aside: formatting strings.

You can include string formatting characters in any string. “\n” for example, is the code for a *newline* (so text following “\n” starts on the next line).

So, if you want to have a message box of this form →

Variations on the following theme would do it.

```
string s1="I want 2 lines of text\n";  
string s2="...And I want to decide the break";  
MessageBox.Show(s1+s2);
```



Coding Conventions

To make our code easier to read and navigate, and thus maintain, we should adopt coding standards that we all follow. Your textbook (Doyle) outlines coding standards that we will adopt in ENGINEER 1D04. We will also add the following convention for naming visual objects:

Visual objects will all have a prefix that identifies their class, followed by a meaningful name. The prefixes to use are as follows:

Button - btn
TextBox - txt
Label - lbl

Some example names include btnGo, btnCalculate, btnClose, btnOkay, txtYear, txtEaster, txtBestDay, lblYear, lblEaster, lblBestDay, etc. The default names provided by Visual C# are not adequate, since they include no hint to the objects purpose. To make your coding easier, the best time to rename the object is immediately after creating it, before you start associating events with it.

In addition to the above, please remember to use comments in your code.

Submission

Zip your project folder to a file named Name_Lxx_Lab4.zip where Name is your name (last name followed by initials – no spaces), and xx is your lab section. Upload this file and submit it as a solution in the Lab Assignments section on WebCT, for “Lab 4.”