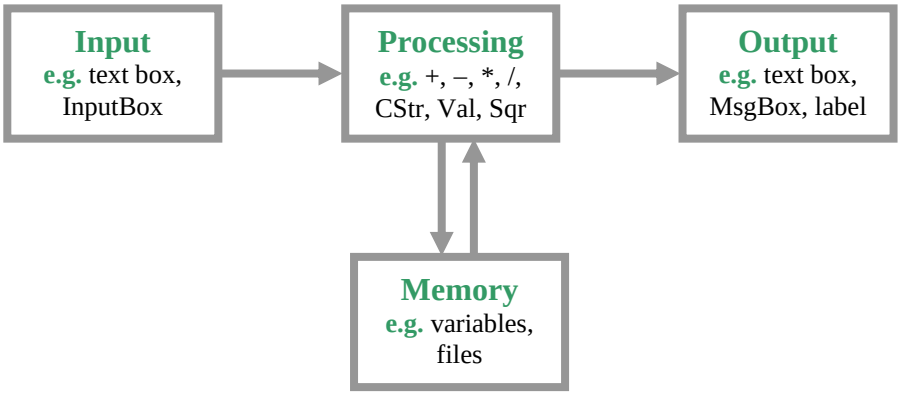
ICS3U0 – Ongoing Review – Practice Problems

Important Points to Consider

What you need to UNDERSTAND to Solve ALL the Problems

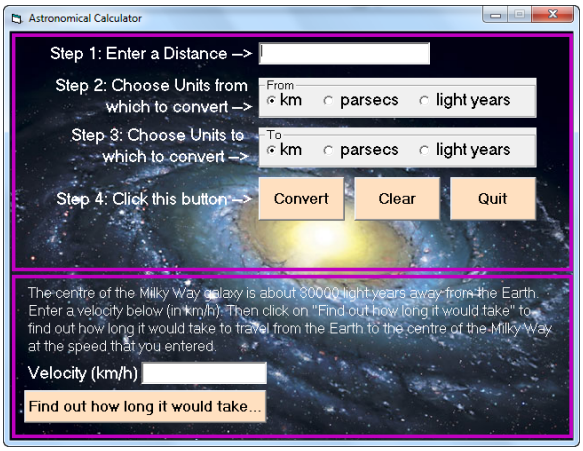


- the most important lesson of the entire course
- sequence, selection, repetition
- numeric operations (+, –, *, /, ^, \, Sqr, **Mod**)
- string operations (we only know "&" so far but will learn more by the time all these problems are solved)
- data types
- data type conversion functions such as "Val" and "CStr"
- variables
- local and global variables
- variable declarations
- numeric and string constants
- defined constants
- objects
- properties and methods
- debugging techniques
- appropriate comments
- proper indentation
- meaningful, descriptive names
- "**If**" statements
- write conditions for "**If**" statements including compound conditions that use "**And**" and/or "**Or**"
- "**Do … Loop Until**," "**Do … Loop While**," "**Do Until … Loop**," "**Do While … Loop**" and "**For … Next**" loops
- string manipulation techniques and functions
- generating pseudo-random integers
- the binary representation of positive integers and the twos complement binary representation of negative integers
- one-dimensional arrays
- dynamically resizing an array (look up "**ReDim**" in MSDN)

PP1 – Astronomical Calculator

1. Copy the folder "**I:\Out\Nolfi\Ics3u0\Practice Problems-Forms without Code\Astronomy**" to your "g:" drive.

2. Give meaningful, descriptive names to all the objects whose names need to be used in the code. (Don't bother renaming the labels whose captions will never change.)



3. This program can be used to perform two different functions.

   The user can enter a distance in kilometres, parsecs or light years, then use option buttons to select the "from units" and the "to units." Write VB code that causes the distance entered by the user to be converted from the "from units" to the "to units" when the "Convert" button is clicked.

   In addition, the user can enter a speed in km/h and then click on the "Find out how long it would take…" button. Write VB code that calculates how long it would take to travel from the Earth to the centre of the galaxy while travelling at the speed entered by the user.

   Note
   speed of light = 299792.458 km/s 1 light year = ***distance*** that light travels in one year 1 parsec = $3.08568025 \times 10^{13}$ km

4. When you are finished, ***copy*** the "**Astronomy**" from your "g:" drive to the folder "**I:\In\Nolfi\Ics3u0*YourName***."

PP2 – WeRCheap Cell Phone Service

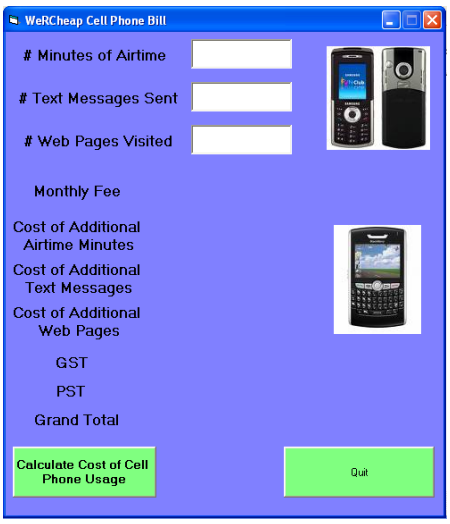
To prevent you from using your cell phone too much, your parents have forced you to subscribe to the following plan from the WeRCheap cell phone company:

Monthly Fee: $10.00

Feature	*Included in Monthly Fee*	*Additional*
Phone Call Airtime	50 minutes	$0.25 per additional minute from 51-100 minutes $0.35 per additional minute over 100 minutes
Text Messaging	50 text messages	$0.05 per additional text message from 51-100 messages $0.15 per additional text message over 100 messages
Web Pages Visited	100 Web pages	$0.05 per additional Web page visited

1. ***Copy*** the folder "**I:\Out\Nolfi\Ics3u0\ Practice Problems-Forms without Code\WeRCheap**" to your "g:" drive.

2. Give meaningful, descriptive names to all the objects whose names need to be used in the code. (Don't bother renaming the labels whose captions will never change.)



3. The user enters the total number of minutes of airtime used, the total number of text messages sent and the total number of Web pages visited.

   Write code to calculate and display the monthly fee, the cost of additional airtime minutes, the cost of additional text messages, the cost of visiting additional Web pages, the GST, PST and the grand total. (Make sure that you use the "Format" function to display the money values with a dollar sign and two decimal places.)

4. Once you are finished, ***copy*** the "**WeRCheap**" folder from your "g:" drive to the folder "**I:\In\Nolfi\Ics3u0*YourName***."

PP3 – Paul Sheppard XBox

Since "Paul Sheppard's Video Game Supply Shop" has become so popular, he is in need of a more efficient method of processing customer orders. Being such a kind, helpful and caring person, you have offered to write a VB program (for a ***very high fee***, of course) that will solve his problem.

Write a Visual Basic program that uses the form shown below to

(1) Input the ***price*** of each item

(2) Input whether there is no tax, GST only, PST only or GST and PST.

(3) Calculate and display the ***totals*** for the entered item (the ***price, PST*** (8%), the ***GST*** (5%)) when the "Enter Price of Next Item" button is clicked.

(4) Calculate and display the ***totals*** for the ***all*** items in the given order (the ***total of all prices, total PST***, ***total GST***) when the "Enter Price of Next Item" button is clicked (you'll need global variables for this).

(5) Calculate and display the ***grand total***.

(6) Clear the previous order and start a new order when the "Start New Order (Clear Previous Order)" button is clicked.

This button needs to be clicked every time a price is entered!

These labels should be used to display information only for the current order. Once the "Start New Order" button is clicked, these labels should be cleared!

Before you begin, copy the folder "**I:\Out\Nolfi\Ics3mo\Practice Problems-Forms without Code\PaulS**" to your "g:" drive.

When you are done, ***copy*** the "**PaulS**" folder from your "g:" drive to

"**I:\In\Nolfi\Ics3mo*YourName*,**"

where *yourName* stands for your name!

PP4 – Income Tax

Every year, Canadians complete income tax returns to calculate how much income tax they must pay. The income tax payable is based on ***taxable income***, which is equal to ***gross income*** minus ***deductions***. In this exercise, you will create a Visual Basic program that calculates the amount of ***federal tax*** payable, the amount of ***provincial tax*** payable, the ***total income tax*** payable and the ***marginal tax rate***.

How Federal and Provincial Income Tax are Calculated

Federal Income Tax (Federal 2004 rates)

- **Tax Bracket 1:** If taxable income is **$31,677 or less**, federal income tax payable is equal to **16% of the taxable income**.
- **Tax Bracket 2:** If taxable income is greater than **$31,677 but not greater than $63,354**, the federal income tax payable is **22% of the *difference* between the taxable income and $31,677, plus $5068**.
- **Tax Bracket 3:** If taxable income is greater than **$63,354 but not greater than $103,000**, the federal income tax payable is **26% of the *difference* between the taxable income and $63,354, plus $12,037**.
- **Tax Bracket 4:** If taxable income is **greater than $103,000**, the federal income tax payable is **29% of the *difference* between the taxable income and $103,000, plus $22,345**.

Provincial Income Tax (Ontario 2004 Rates)

- **Tax Bracket 1:** If taxable income is **$31,893 or less**, provincial income tax payable is equal to **6.05% of the taxable income**.
- **Tax Bracket 2:** If taxable income is **greater than $31,893 but not greater than $63,786**, the provincial income tax payable is **9.15% of the *difference* between the taxable income and $31,893, plus $1930**.
- **Tax Bracket 3:** If taxable income is **greater than $63,786**, the provincial income tax payable is **11.16% of the *difference* between the taxable income and $63,786, plus $4,848**

How Marginal Tax Rate is Calculated

(Marginal Tax Rate) = (Total Income Tax Payable) ÷ (Taxable Income) × 100%

Examples

Taxable Income	*Federal Income Tax Payable*	*Provincial Income Tax Payable*	*Total Income Tax Payable*	*Marginal Tax Rate*
$30,000	$30,000 × 0.16 = **$4800**	$30,000 × 0.0605 = **$1,815**	$4,800 + $1,815 = **$6,615**	$6,615 ÷ $30,000 × 100% = **22.05%**
$50,000	($50,000 – $31,677) × 0.22 = $4,031.06 $4,031.06 + $5,068.00 = **$9,099.06**	($50,000–$31,893) × 0.0915 = $1,656.79 $1,656.79 + $1,930.00 = **$3,586.79**	$9,099.06 + $3,586.79 = **$12,685.85**	$12,685.85÷$50,000×100% = **25.37%**
$80,000	($80,000–$63,354) × 0.26 = $4,327.96 $4,327.96 + $12,037.00 = **$16,364.96**	($80,000–$63,786) × 0.1116 = $1,809.48 $1,809.48 + $4,848.00 = **$6,657.48**	$16,364.96+ $6,657.48 = **$23,022.44**	$23,022.44÷$80,000×100% = **28.78%**
$150,000	($150,000–$103,000)×0.29 = $13,630 $13,630.00 + $22,345.00 = **$35,975.00**	($150,000–$63,786)×0.1116 = $9,621.48 $9,621.48 + $4,848.00 = **$14,469.48**	$35,975 + $14,469.48 = **$50,444.48**	$50,444.48÷$150,000×100% = **33.63%**

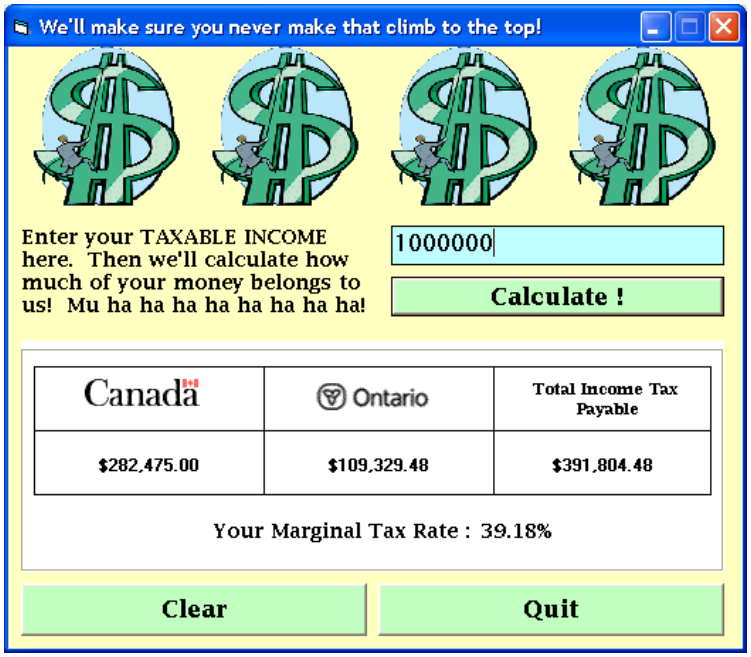
Exercises

1. Using the above examples as a guide, complete the following table.

Taxable Income	*Federal Income Tax Payable*	*Provincial Income Tax Payable*	*Total Income Tax Payable*	*Marginal Tax Rate*
$63,700				

2. Copy the folder "**I:\Out\Nolfi\Ics3u0\Practice Problems-Forms without Code\Income Tax**." Load the VB program "**I:\Out\Nolfi\Ics3u0\ Practice Problems-Forms without Code\Income Tax\IncomeTax.vbp**." The design of the form is done for you. Your job is to write code that will allow the program to calculate the federal income tax payable, provincial income tax payable, total income tax payable and the marginal tax rate. All displayed values should be rounded to two decimal places and appropriately formatted.

Note: Once you load the program into VB, you will notice that a small amount of code has already been provided for you. Do not delete or modify the code. It has been included to help prevent run-time errors. In addition, do not modify the form in any way. This will simply cause you to waste time.



3. When you are finished, ***copy*** the "**Income Tax**" folder from your "g:" drive to the folder "**I:\In\Nolfi\Ics3u0*YourName*.**"

PP5 – Newfoundlander Style Pizza

Since Tyler is so busy kneading the dough for his Newfie Screech Style Pizza, he does not have much time to process customer orders. Therefore, he is seeking your help! His restaurant, ***Newfie Screech Style Pizzeria***, needs a computer program that can process customer orders.

As shown in the table, there is a base price for each pizza, plus an additional charge for each topping.

Size	Base Price	Each Topping
Small	$9.95	$1.00
Medium	$12.95	$1.25
Large	$15.95	$1.50
Party Size	$18.95	$2.00
Drinks	$1.25	

Write a Visual Basic program that uses the form shown below to

(1) Input the ***size*** of the pizza, the ***number of toppings***, the ***number of pizzas*** and the number of ***drinks***

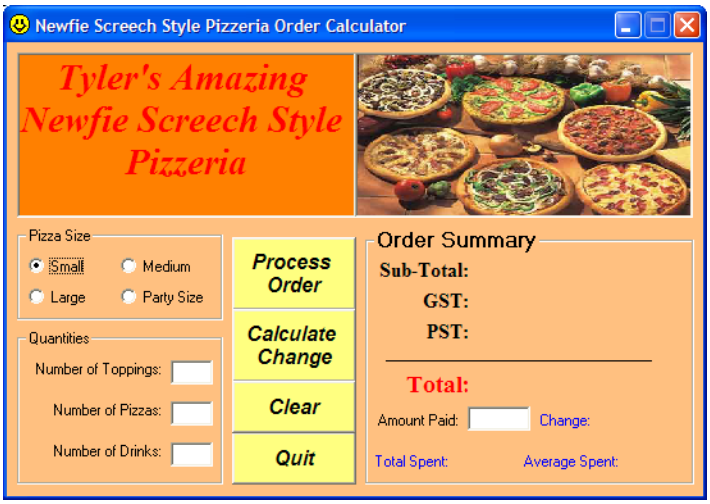
(2) Calculate and display the ***sub-total*** (cost before tax), the ***PST*** (8%), the ***GST*** (7%) and the ***total***

(3) Input the ***amount of money paid*** by the customer

(4) Calculate and display the ***change*** that the customer should receive

(5) Calculate and display the ***total*** amount spent by all customers

(6) Calculate and display the ***average*** amount spent by each customer.



Before you begin, copy the folder "**I:\Out\Nolfi\Ics3mo\Practice Problems-Forms without Code\Newfie Pizza**" to your "g:" drive.

When you are done, copy the "**Newfie Pizza**" folder from your "g:" drive to

"**I:\In\Nolfi\Ics3mo*YourName*,**"

where *yourName* stands for your name!

Use the table on the next page to plan your solution in a logical, organized fashion

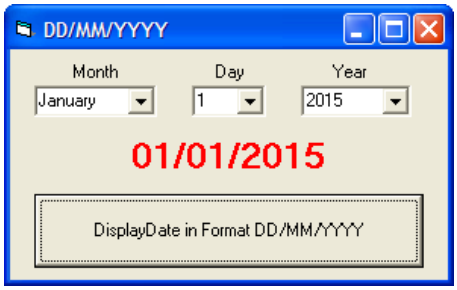
Planning the Solution in a Logical, Organized Fashion

INPUT	PROCESSING	OUTPUT
What information does the user enter?	What must be done with the information?	What should be displayed after processing is complete?
Code for Input	**Code for Processing**	**Code for Output**
VARIABLES (MEMORY)		
LOCAL VARIABLES		**GLOBAL VARIABLES**

PP6 – Date and Time

Part 1

Copy the contents of the folder "**I:\Out\Nolfi\Ics3u0\Practice Problems-Forms without Code\Date**" to your "g:" drive. Within this folder you will find a VB project file called "**Date.vbp**." Load the "Date.vbp" project and experiment with it for a few minutes. You will discover that three ***combo boxes*** are used to allow the user to select the month, day and year. (A combo box combines the functionality of a text box with that of a list box.)



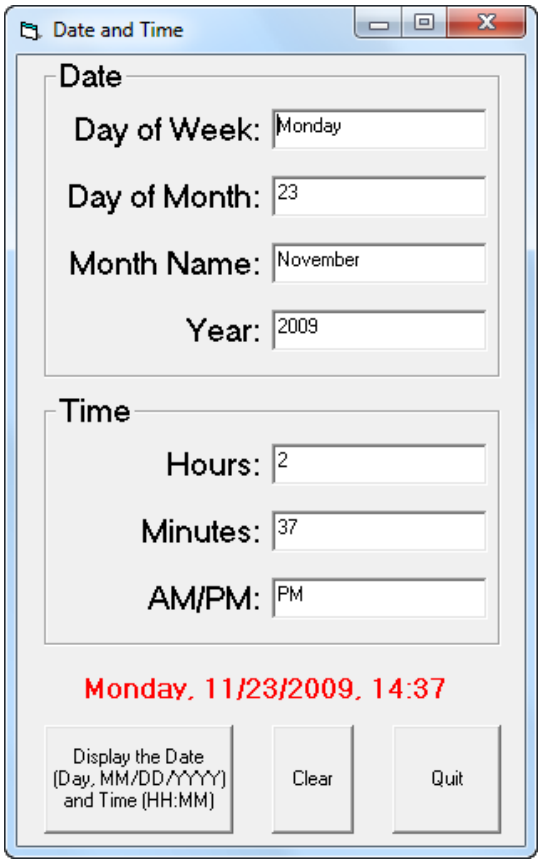
When you examine the VB code for this project, it may look very complicated to you. ***Please do not be discouraged by the appearance of the code!*** All you need to do is write the code for the command button. That is, you must write code that takes the date given by the values stored in the combo boxes and converts it to the format DD/MM/YY (2 digits for the day, 2 digits for the month and four digits for the year).

Note: Although it is not required at this point, students who are confident enough may wish to study the code given in this project. Since this program contains a plethora of new ideas to explore, it is possible to learn a great deal from it!

When you are finished, ***copy*** the "**Date**" folder to "**I:\In\Nolfi\Ics3u0*YourName***."

Part 2

Copy the contents of the folder "**I:\Out\Nolfi\Ics3u0\Practice Problems-Forms without Code\Date and Time**" to your "g:" drive.



As shown in the example form at the right, the user enters

- the day of the week in words (e.g. Monday)
- the day of the month (e.g. 23)
- the month name (e.g. November)
- the year (e.g. 2009)
- the time (separate text boxes are used to enter the number of hours, the number of minutes and whether the time is AM or PM)

The program then displays a single string consisting of

- the day of the week in words followed by a comma and a space (e.g. Monday,)
- the date in the format MM/DD/YYYY followed by a comma and a space (e.g. 11/23/2009,)
- the time in ***24 hour clock format*** (e.g. 14:37)

In the given example, the entire string would look as follows:

Monday, 11/23/2009, 14:37

When you are finished, ***copy*** the "**Date and Time**" folder to "**I:\In\Nolfi\Ics3u0*YourName***."

Note

You may prefer to use combo boxes instead of text boxes for this program.

PP7 – Nolfi's ChangeMaker

Description of Problem

Write a VB program that takes a value specified in dollars and cents and then displays the number of "toonies," "loonies," quarters, dimes, nickels and pennies that are required to equal the entered amount.

Hint: This problem is almost identical to the "time converter" problem. The **"Mod"** and **"\"** operators should prove to be ***extremely helpful!***

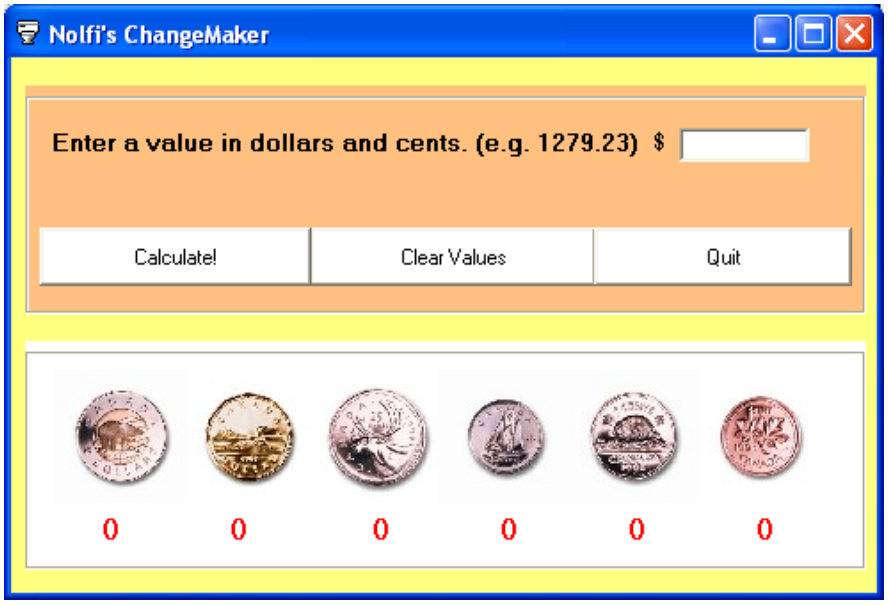
You will find a form for this program in "**I:\Out\Nolfi\Ics3u0\Practice Problems-Forms without Code\ChangeMaker**"

When you are finished, ***copy*** the "**ChangeMaker**" folder to "**I:\In\Nolfi\Ics3u0*YourName***."

A Note on Evaluation

The label boxes that appear directly beneath the images of the coins are organized as a ***control array*** of label boxes called "`lblNumCoins`." The label box beneath the "toony" is "`lblNumCoins(0)`," the label box beneath the "loony" is "`lblNumCoins(1)`" and so on. This allows you to write this program using a **"For … Next"** loop.

- For a ***level 4 mark***, you must use a **"For … Next"** loop along with a control array.
- If you do not understand control arrays, it will not be possible to use a loop to solve this problem. However, you may still earn a ***high level 3 mark*** by deleting the control array of label boxes and using individually named label boxes.



PP8 – Leap Year Practice Problem

Background Information

The ***calendar year*** is 365 days long, except in a ***leap year***, in which case an extra day is added to February to make the year 366 days long. A given year is a leap year if ***both*** of the following conditions are satisfied:

1. The year is divisible by 4.
2. If the year is divisible by 100, then it must also be divisible by 400.

The second condition above applies to years at the end of a century, for example, 1800, 1900 and 2000. Although such years are divisible by 4, they are ***leap years*** only if they are exactly divisible by 400. Therefore, for instance, 1800 and 1900 ***were not*** leap years but 2000 was. The reason for these rules is to bring the ***average length*** of the calendar year into line with the length of the Earth's orbit around the Sun. This ensures that the seasons always occur during the same months each year. Note that the rules used for the Gregorian calendar make the adopted average length of the year 365.2425 days, a slight deviation from the actual length of the ***tropical year***, 365.24219 days. This slight deviation produces a seasonal shift of about three days in 10000 years.

The rules stated above have been in use since 1582, when Pope Gregory XIII instituted the ***Gregorian calendar***. Prior to 1582, the ***Julian calendar*** had been used in the Western world. It was established in 46 BC by Julius Caesar, who was the emperor of the Roman Empire at the time. In the Julian calendar, each year consisted of 12 months and it was assumed that there were an average of 365.25 days in a year. This was achieved by having three years consisting of 365 days and one year consisting of 366 days. The discrepancy between the actual length of the year, 365.24219 days, and the adopted length, 365.25 days, may not seem important at first glance but over hundreds of years, the difference becomes obvious. The reason for this is that the seasons become progressively out of kilter with the calendar date using the Julian calendar.

Statement of the Problem to be Solved

Write a Visual Basic program that can

(a) determine whether a given year is a leap year
(b) calculate the seasonal shift produced by the rules of the Julian calendar given the number of years
(c) calculate the seasonal shift produced by the rules of the Gregorian calendar given the number of years

For example, your program would produce the following outputs given the inputs shown below:

Input	*Output*
1900	1900 was not a leap year
2020	2020 will be a leap year
10000	The rules of the Gregorian calendar produce a seasonal shift of about 3.1 days.
10000	The rules of the Julian calendar produce a seasonal shift of about 78.1 days.

Evaluation

A good solution will have the following characteristics:

(a) output is correct for all inputs
(b) code is indented properly
(c) blank lines are inserted in strategic places to make code easier to read
(d) **Option Explicit** is used
(e) ***descriptive identifier names*** are used (for variables, constants, objects, etc) and ***naming conventions*** are followed (e.g. "cmd" prefix used for command button names, "CamelCase," etc)
(f) comments are used to explain abstruse (difficult-to-understand) code
(g) comments are used to introduce major blocks of code
(h) the user interface (i.e. your form) is well designed and user-friendly

What to do when you are Finished

Save a copy of your program in "**I:\In\Nolfi\Ics3u0*YourName***," where *YourName* stands for your name.

PP9 – Mathematical Practice Problems

1. A number is called ***perfect*** if the sum of its ***proper divisors*** is equal to the number itself. Two examples of perfect numbers are 6 and 28 because 6 = 1 + 2 +3 and 28 = 1 + 2 + 4 + 7 + 14. Write a program that finds ***all*** perfect numbers within the range of an **Integer** variable.
2. Write a program that finds the ***greatest common divisor*** of any two integers. For example, the greatest common divisor (GCD) of 24 and 40 is 8.
3. Write a program that finds the ***least common multiple*** of any two integers. For example, the least common multiple (LCM) of 24 and 40 is 120.
4. Write a program that can add, subtract, multiply and divide two fractions and display the result in lowest terms.
5. Write a program that can factor a ***simple trinomial***, if it is factorable. If the simple trinomial is not factorable, the program should display a suitable message
6. Write a program that will convert any **Long** integer specified in ***decimal*** (base ten) form to ***binary*** (base 2) form. Note that to convert ***negative*** integers to binary form, you must understand something called the "twos complement" binary representation of negative integers. You will receive extra credit for learning about the "twos complement" representation without the teacher's help.
7. The numbers 220 and 284 are called an ***amicable pair*** because the sum of the proper divisors of 220 is 284 and the sum of the proper divisors of 284 is 220. Write a program that finds ***all*** amicable pairs within the range of an **Integer** variable.

PP10 – Array Practice Problems

1. Fill an array of ten elements ***at random*** with the numbers 0 to 9 with ***no repetition***.
2. Fill an array of ten elements ***at random*** with *ten different* two-digit numbers.
3. Fill an array of ten elements ***at random*** with the numbers 1 to 5 so that each number appears ***exactly twice***.
4. Display all possible ways to choose a pair of numbers from a list of ten numbers. Note that order does not matter in this problem. For example, the pair "2, 5" is considered the same as the pair "5, 2."
5. Remove all even numbers from an array of numbers.
6. Split an array of even numbers into two arrays, one of ***even*** numbers and one of ***odd*** numbers.
7. Remove any repeated numbers from an array without leaving any gaps.
8. Determine the ***mean*** (average) of an array of numbers.
9. Determine which number(s) occur most frequently in an array of numbers (i.e. find the ***mode***).
10. Determine the ***median*** of an array of numbers. That is, find the number that divides the array into two halves, one half consisting of all the numbers ***less than*** the median and the other half consisting of all the numbers ***greater than*** the median.
11. Remove all the spaces from a string.
12. Remove all the vowels from a word.
13. Scramble the letters in a word.
14. Write a program that randomly generates a set of six integers ranging from 1 to 49 (for Lotto 6/49). The numbers should be displayed in ascending order (smallest to largest). Do not forget to use the Internet to search for relevant images!

PP11 – French Conjugation

Write a program that conjugates any regular French verb. If you need to brush up on your French grammar, here are the correct endings for the three types of regular verb infinitives:

-er verbs	-re verbs	-ir verbs
e	s	is
es	s	is
e	-	it