



Course number : 420-P25-AS

## INTRODUCTION TO OBJECT PROGRAMMING

Teacher : Maftai Mihai

Weighting : **30%**  
 Points number: **100**  
 Duration :

Group : **07054**  
 Date : **2018-07-26 before 10:00**  
 Session : **Summer 2018**

### STATEMENT OF THE COMPETENCY

- To use object-oriented development approach (016T)

### REQUIRED COMPETENCIES FOR THE PROJECT

1. To create an object model
2. Refine the object model.
3. To program a class
4. To ensure that the class functions correctly

### DIRECTIVES

- **Open book and notes.**

### INSTRUCTIONS

**This project has 6 sections**

The application has 5 Forms, and 1 document evaluated like this:

| <b>Form 0</b>   | <b>Form 1+2</b>             | <b>Form 3</b>        | <b>Form 4</b>       | <b>Form 5</b>             | <b>Document</b>                  | <b>Total</b> |
|-----------------|-----------------------------|----------------------|---------------------|---------------------------|----------------------------------|--------------|
| Dash Board      | Lotto Max<br>+<br>Lotto 649 | Simple<br>Calculator | Money<br>Conversion | Temperature<br>Conversion | Describing<br>the<br>application |              |
| <b>5 points</b> | <b>20 points</b>            | <b>20points</b>      | <b>20points</b>     | <b>20points</b>           | <b>15 points</b>                 | <b>100 p</b> |

### Form 0 - The Dash board (5 points)

Create a C# Form application similar with the one in the images below:



The Exit button will confirm before exiting.

Identify yourself, enter the current date and have a short description for the application as a comment. By clicking on one of those buttons the application will open one of the following form applications:

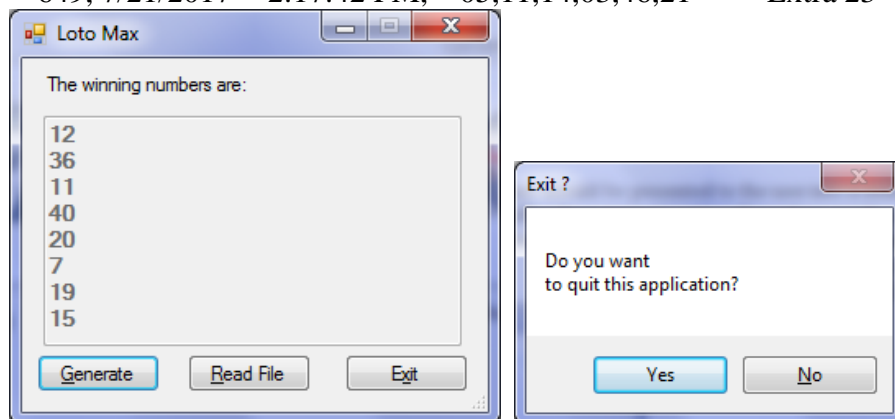
### Form 1 + 2- The Lotto Max and 649 application (20 points)

The form application that allows the user to generate 7 + 1 or 6 + 1 random numbers out of 49 for the game Lotto Max or Lotto 649.

```
Random random = new Random();
int randomNumber = random.Next(1, 49);
```

The 8 or 7 unique numbers should be presented to the user into a multiline read only TextBox and also saved into a text file **LottoNbrs.txt** (one series of numbers per line, using comma to separate the numbers). Identify the name of the lottery and add the current date and time. Ex.:

Max, 7/21/2017 2:17:36 PM, 40,33,12,06,36,04,18 Extra 29  
649, 7/21/2017 2:17:42 PM, 05,11,14,03,46,21 Extra 23



Read and show the text file content using a message box. Add pertinent information into the message box and add the appropriate title + your name as title.

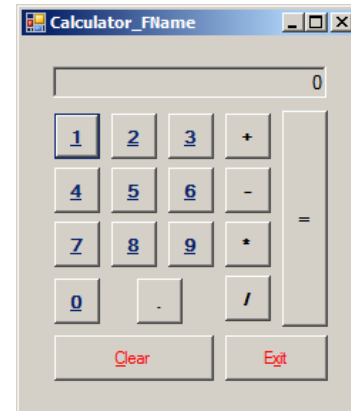
Use: FileStream, StreamReader, StreamWriter objects

### Form 3 - Simple calculator application (20 points)

You'll design a form that lets the user perform the operations provided by a basic calculator. You'll also create a class that performs the required basic operation, and presenting the result into the read only TextBox. Also save all the arithmetical operations and theirs results into a text file **Calculator.txt** (one operation per line).

Ex.: You have to convert entry data back and fore between decimal or double (data type for calculations) to the string data type (for presentation).

So, by clicking on one button from 0 to 9, you want to present the first or the second number into the text box, and also, when you click on one of those 4 operation buttons, you need to have that value as a number to be used in further calculations, and when the equal button is pressed, the result of the operation will be displayed. Clear button will clear all the fields' members of the Calculator class, and the text box.



- If you want to be more specific about the Calculator class, you can provide this class design:

| Private field                 | Description  |
|-------------------------------|--|
| <b>currentValue</b>           | A decimal that stores the result currently displayed by the calculator.  |
| <b>operand1</b>               | A decimal that stores the value of the first operand.  |
| <b>operand2</b>               | A decimal that stores the value of the second operand.   |
| <b>op</b>                     | An string type that stores the value of the operator   |
| Constructor                   | Description  |
| <b>( )</b>                    | Creates a Calculator object with default values. The default value for the op field is Null.   |
| Property                      | Description  |
| <b>CurrentValue</b>           | Gets the value of the currentValue field.  |
| Method                        | Description  |
| <b>Add(displayValue)</b>      | Sets the operand1 and currentValue fields to the value that's passed to it, and sets the op field to "+".  |
| <b>Subtract(displayValue)</b> | Sets the operand1 and currentValue fields to the value that's passed to it, and sets the op field to "-".  |
| <b>Multiply(displayValue)</b> | Sets the operand1 and currentValue fields to the value that's passed to it, and sets the op field to "*".  |
| <b>Divide(displayValue)</b>   | Sets the operand1 and currentValue fields to the value that's passed to it, and sets the op field to "/".  |
| <b>Equals()</b>               | Performs the operation specified by the op field on the operand1 and operand2 fields, and stores the result in the operand1 field.   |
| <b>Equals(displayValue)</b>   | Sets the operand2 field to the value that's passed to it. Then, performs the operation specified by the op field on the operand1 and operand2 fields, and stores the result in the operand1 field. |
| <b>Clear()</b>                | Sets the private fields to their default values.   |

Use: FileStream, StreamReader, StreamWriter objects

## Form 4 - The Money Exchange application (20 points)

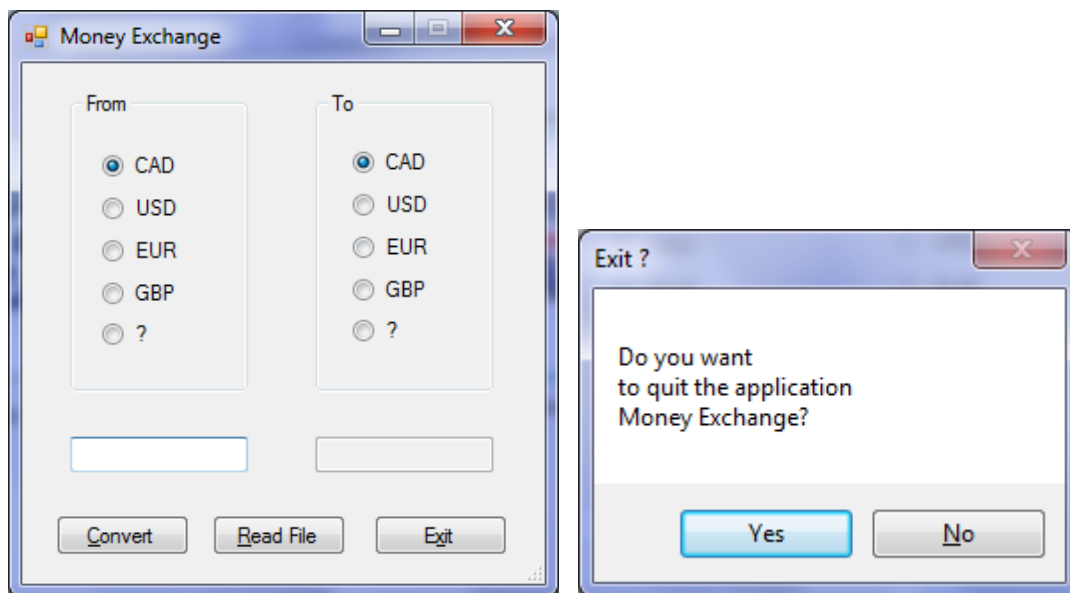
Add another currency of your choice, use the exchange factor , you find it on the Internet (save it as a comment + the date).

The choice of 4 + 1 exchange currencies, should be presented by using radio buttons and allow the user to enter a valid amount into a text box (left), and presenting the converted amount into the read only TextBox (right).

Save the conversion into a text file **MoneyConversions.txt** (one conversion per line. Identify the currency from/ to and add the current date and time. Ex.:

100 CAD = 78 USD, 4/21/2017 9:57:33 AM

100 EUR = 88 USD, 4/21/2017 10:07:03 AM



Use: Try and Catch, Regular Expressions and FileStream, StreamReader, StreamWriter objects.

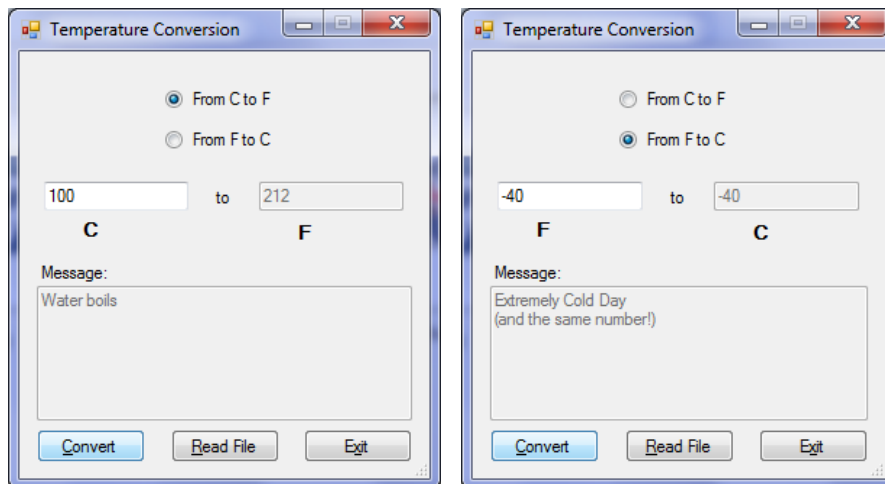
Read and show the text file content using a message box. Add pertinent information into the message box and add the appropriate title + your name as title.

### Form 5 - The Temperature application (20 points)

The choice of 2 possible temperature conversions should be presented to the user by using radio buttons and allow him to enter the valid value into a text box, and presenting the corresponding temperature into the read only TextBox. Into the Message read only text box, write the appropriate message description (see the table on the next page). Also save the temperature conversion into a text file **TempConversions.txt** (one conversion per line. Identify the conversion from/ to and add the current date and time. Ex.:

100 C = 212 F, 3/22/2017 01:01:33 PM

104 F = 40 C, 3/22/2017 10:07:03 PM



Read and show the text file content using a message box. Add pertinent information into the message box and add the appropriate title + your name as title.

Use try and catch to validate data, and show clear messages to the user.

### Formula for temperature conversion:

Celsius to Fahrenheit:  $(^{\circ}\text{C} \times \frac{9}{5}) + 32 = ^{\circ}\text{F}$

Fahrenheit to Celsius:  $(^{\circ}\text{F} - 32) \times \frac{5}{9} = ^{\circ}\text{C}$

### Examples:

| $^{\circ}\text{C}$ | $^{\circ}\text{F}$ | Description                                  |
|--------------------|--------------------|--|
| <b>100</b>         | <b>212</b>         | Water boils                                  |
| <b>40</b>          | <b>104</b>         | Hot Bath                                     |
| <b>37</b>          | <b>98.6</b>        | Body temperature                             |
| <b>30</b>          | <b>86</b>          | Beach weather                                |
| 21                 | 70                 | Room temperature                             |
| <b>10</b>          | <b>50</b>          | Cool Day                                     |
| <b>0</b>           | <b>32</b>          | Freezing point of water                      |
| -18                | 0                  | Very Cold Day                                |
| <b>-40</b>         | <b>-40</b>         | Extremely Cold Day<br>(and the same number!) |
| (bold are exact)   |                    |  |

Use: Try and Catch, Regular Expressions and FileStream, StreamReader, StreamWriter objects.

Create a document with print screens of your application, also present all the classes , and methods, that you use to build this application.**(10 points)**

Test your application functionalities with different sets of data, save the solution, compressit and send it with the results text files by LEA of Omnivox.

Thank you.