

**Lebanese International University – All Campuses**

**School of Arts and Sciences - Department of Computer Science**

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name** | : Visual Programming | **Course Code** | : CSCI370 |
| **Date** | : Saturday May 2nd, 2020 | **Section** | : |
| **Instructor** | : | **Time** | : 07:00 – 23:59 |
| **Auditorium** | : | **Seat Number** | : |

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of pages** | : 09 | **Allowed Time** | : 75 minutes |
| **Documents** | : | **Calculators** | : Allowed |

|  |  |  |
| --- | --- | --- |
| **Problem#** | **Grade** | **Total Grade** |
| Part I |  | /100 |
| Question 1 | /15 |
| Question 2 | /35 |  |
| Question 3 | /50 |  |

**Guidelines:**

* **Create a VS studio project with your name**
* **Upload to classroom**

**Good Luck**

**Part I**

We will be implementing an application that allows adding and searching for insurance policies.

**Question 1: [15 pts]**

We start by creating an “Insurance” class that has the following properties:

* Type : string
* Customer : string
* Phone : string
* PolicyAmount : int
* Years : int

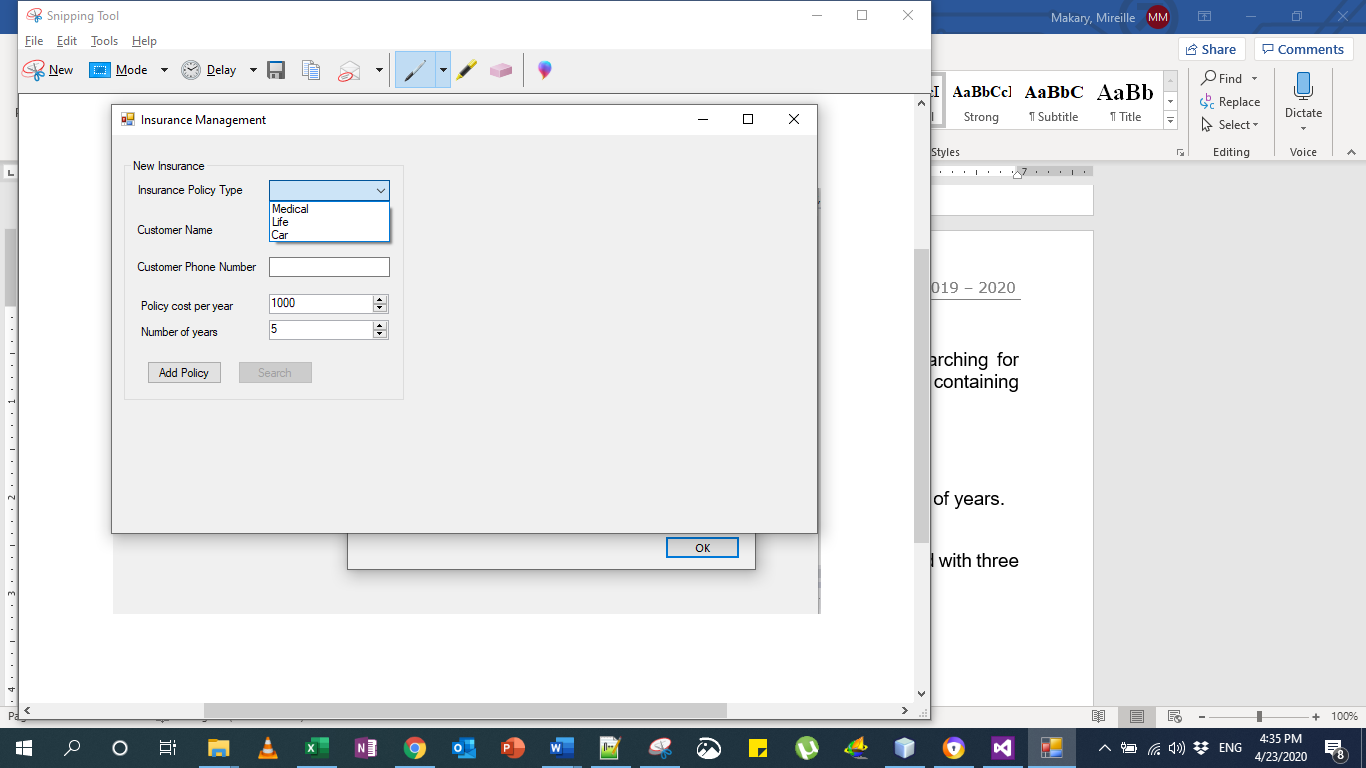
We define the necessary properties, constructor and a ToString method that shows “Insurance Details: Type,Customer,Phone,Total" where Total=PolicyAmount\*Years.

**Question 2 – New Insurance [35 pts]:**

When the application starts, it shows a **Groupbox** containing the below controls:

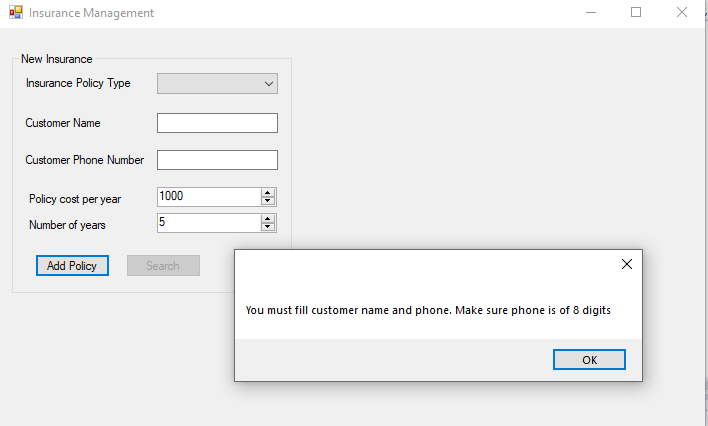
* Five Labels
* A combo box for the policy type which can be: Medical, Life or Car
* Two textboxes for the customer name and phone. **The phone text box allows entering digits only.**
* Two numericupdown controls for the policy amount and number of years.
  + Policy amount ranges between 1000 and 10000, the increment step is 200 **[3pts]**
  + Number of Years5 ranges between 1 and 5, the increment step is 1 **[2pts]**.
* Two buttons to add a policy and for searching.

When the form first loads, the combo box for the policy type will be filled with three options: **Medical, Life and Car**.



The “Search” button is initially **disabled**, **until the user fills the first policy**. Then it becomes enabled.

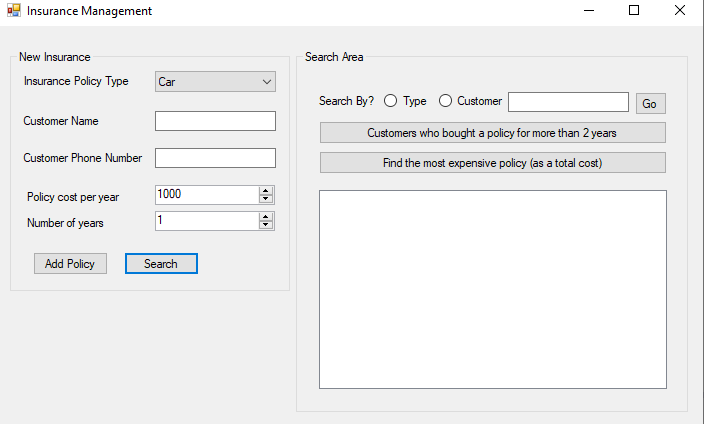
If the user clicks the “Add Policy” button without filling the necessary fields **or in case the phone number had more or less than 8 digits, you can use the Length property of strings to test**, the below message will be displayed



If all the entries are valid, a new Insurance policy object is created and added to an array list of type “Insurance”.

Write the necessary code to:

1. Load the combo box with the 3 type values. [5pts]
2. Allow only digit inputs in the phone text box [5pts]
3. Validate the entries [5pts]
4. Add a new policy when the button Add is clicked, clear and reset all fields. [10 pts]
5. Enable the search button after the first policy and upon clicking the button Search the Groupbox “Search Area” becomes visible as shown in the figure below [5 pts]



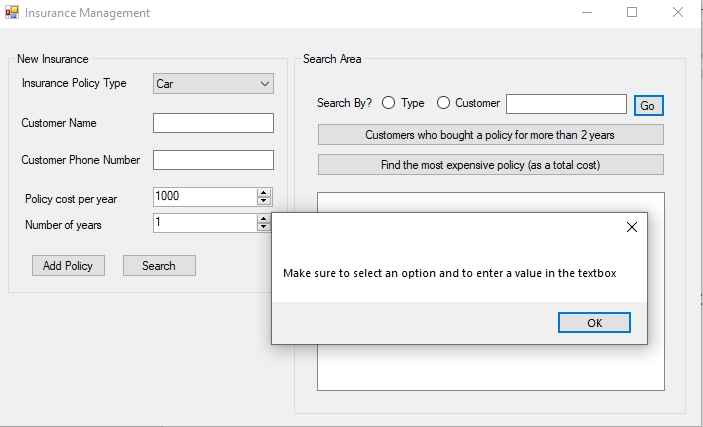
**Question 3 – Search Area [50 pts]**

The Search Area groupbox consists of:

* One label : « Search By ? »
* Two radio buttons
* A textbox
* Three buttons
* A listbox to show the results

**You can implement the code using LINQ queries or the Query Methods and Lambda Expressions.**

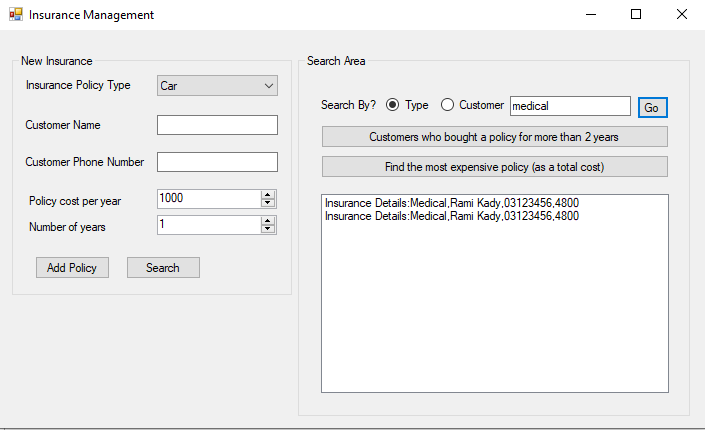
If the user clicks on the button “Go” without selecting any radio button or entering a filtering value in the text box, a message is displayed as shown below:

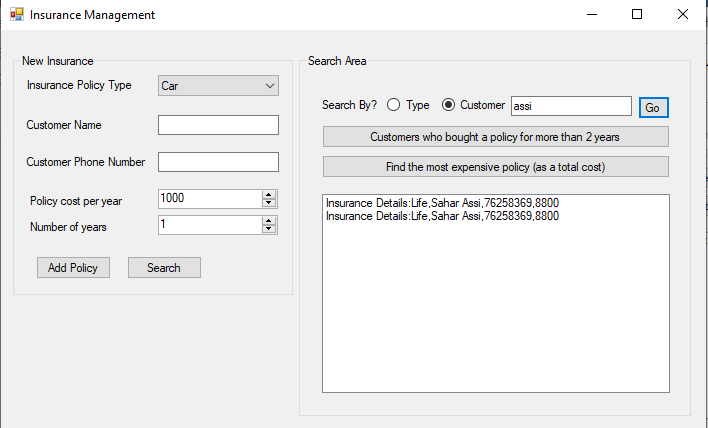


If the user selected “Type”, this means that he will be searching for policies by the type entered in the textbox. If the user selected “Customer”, then he will be looking for the policies belonging to a customer with a name containing the value entered by the user. The results are to be displayed in the listbox.

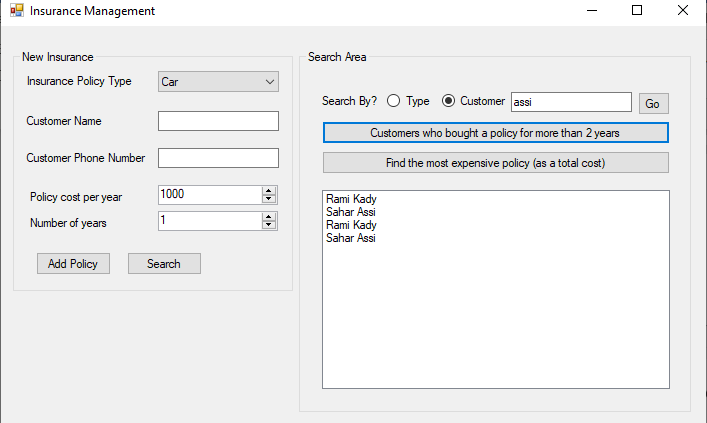
**Hints:**

* When comparing between two strings, to avoid a mismatch because of the cases: Medical and medical for example, we can convert both to lower before testing:
  + String x=”Medical”, y=”medical”;
  + ***x.ToLower().Equals(y.ToLower()) 🡪 returns true.***
* Contains method can be used with Strings to test if a string is a substring of another. ***“Rami Kady”.ToLower().Contains(“rami”.ToLower()) 🡪 yes***





1. Implement the « Go » button as explained [20 pts].
2. The button with text: Customers who bought a policy for more than 2 years, will show in the list box, the customer names only for those who have a Years property > 2. Write the necessary code [15 pts]



1. The last button shows in the listbox the most expensive policy, that is the one that has the highest total cost: PolicyAmount \* Years. Write the necessary code. [15 pts]

