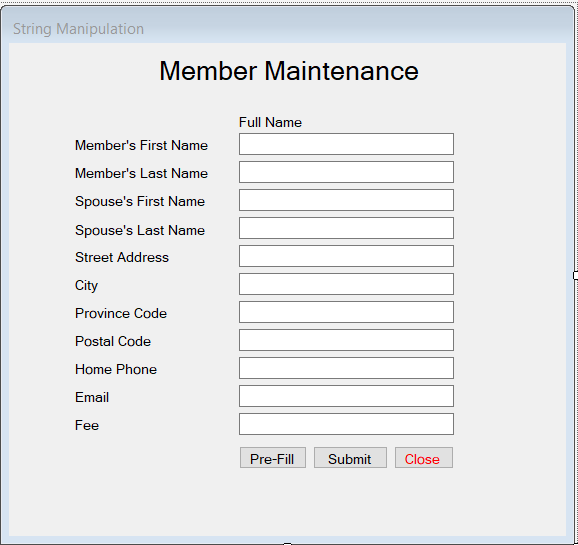
# Programming Concepts II – Assignment 2 – String Manipulation

This assignment is an exercise in using string manipulation to edit and reformat input fields. Some fields have patterns, some expect numeric values … possibly with limits. Some fields are required (and cannot be just spaces), some are optional or conditionally optional, some will cause abends if null is not expected. Strings need to be trimmed of leading and trailing blanks, shifted to upper- or lower-case or capitalized.

1. Create the following form, naming it ***XXMember.cs***, where “XX” are your initials.



1. Add the following methods to either the code for this form or a separate *XXValidation* class:
   1. *XXCapitalize* – this takes a string parameter and returns a string. It should convert an incoming null string to an empty one, trim it of leading & trailing spaces, shift it to lower case and then capitalise each word in the string. Beware of 1-letter words.
   2. *XXPostalCodeValidation* – this takes a string and returns a Boolean. If the incoming string is null or an empty string, pass it. Otherwise, return true only if the string’s entire content fits the Canadian postal pattern “A2A 2A2” … accept upper or lower case and with or without the single space.
   3. *XXPhoneNumberValidation* – this takes a string and returns a Boolean. If the incoming string is null or empty, pass it. Otherwise, return true only if the entire string fits the phone pattern 123-123-1234, with or without the dashes.
   4. *XXIsNumeric* – this takes a string and returns a Boolean. If the incoming string is null or empty, return false. Return true only if the string contains a number defined as: at least one digit, an optional leading dash and an optional single decimal place anywhere in the number.
2. Buttons:
   1. Pre-Fill” loads the form with data … making it easier to test individual edits
   2. “Submit” edits and reformats the fields (back onto the form).
      1. All validation errors are displayed at once in a red label on the form or in a message box … one line per error, focus to first field in error.
   3. “Close” closes the form.
3. Perform the following validations (all fields are optional unless specified otherwise). All error messages are to be shown at once, one line per error, focus to first field in error.
   1. The member’s first name and last name are required and cannot be just blanks.
   2. If an email is provided, the postal information (street address, city and postal code) is optional. If email is not provided, the postal information is required. All four are fine, but not mandatory.
   3. The province code, if provided, must be exactly two **letters**, not just two characters. Accept upper- and lower-case.
   4. Use your *XXPostalCodeValidation* method to verify the postal code.
   5. Use your *XXPhoneNumberValidation* method to verify the phone number.
   6. The email must be a valid email pattern if provided. There’s a MailAddress class in the System.Net.Mail namespace that throws an exception when instantiated with an invalid email address. There’s also a plethora of indecipherable regular expressions out there.
   7. Use your *XXIsNumeric* method to verify the fee, then ensure it is not less than zero.
4. Perform the following data conversions
   1. Use your *XXCapitalize* method to capitalise the member’s and spouse’s names, the street address, and the city.
   2. Derive *Full Name* from the member’s and spouse’s names (after capitalization). Consider these examples for Shania Twain and Russell Crowe:
      1. If there’s no spouse, it’s the member’s last and first names, with a comma:

*Twain, Shania*

* + 1. If the spouse’s last name is not provided or is the same as the member’s:

*Twain, Shania & Russell*

* + 1. If the spouse’s last name is different:

*Twain, Shania & Crowe, Russell*

* 1. Shift the province code to upper case.
  2. Shift the postal code to upper case and insert a single space if there isn’t one already.
  3. Insert the dashes into the phone number, if they are not there already.
  4. Shift the email to lower case (easier to use for comparisons).
  5. Display the fee with two decimal places … rounded, if necessary … no commas or currency symbols.

## Hand In

1. Zip and upload your project folder to the assignment 2 drop-box for this course.
2. Hand the marking sheet in during class or in the drop-box on your teacher’s office door.