Assessed Exercise 2 : Binary SearchTree

(weeks 4,5)

Create a **C# windows form application** to maintain a binary search tree of type **int**. Your windows application should allow

1. a new item (int) to be inserted in the tree (**Hint**: refer to week6 lecture note page5, 9 for creating node class and binarytree class, page 11 for inserting new item / lab note Exercise1)
2. total number of items in tree to be displayed (**Hint**: refer to lab note Exercise1 Count())
3. items in tree to be traversed using PreOrder, InOrder and PostOrder and displayed (**Hint**: refer to week6 lecture note page8, 9 / lab note Exercise1)
4. Alter your tree and windows form to store **Customer** objects (re-use the class from Assessed Exercise 1) and ammend your form to allow customer details to be entered. The number of items in the tree should be retained. (**Hint**: Implement a class Customer – refer to week3 lecture note page10-12, but need to use IComparable interface for comparing two objects (see the following hints))
5. Add functionality to allow retrieval of a customers details by providing a name.

This exercise will be graded from 0-100%, a good attempt at 1,2 only (with demonstrable understanding only could pass, completion of 1-3 only is worth up to 60%

**Hint** : To implement 4 (in a more professional manner) your Customer class should implement an interface called [IComparable](https://learn.microsoft.com/en-us/dotnet/api/system.icomparable.compareto?view=net-8.0), which allows two objects of a class to be greater, less or equal to each other (need for a BinaryTree class). To add this to your Customer class, alter the class header as below :

class Customer : IComparable //implement method CompareTo

You should see an error – red line under IComparable as we haven’t added a CompareTo method. Hover the mouse over “IComparable” and an info box will appear – click on “show potential fixes”, then click on “implement interface through name”, which will add the CompareTo method, using the name member to order Customer objects. Inside of the CompareTo function, remember to cast objects to Customer type. Example:

Customer other = (Customer)obj; // cast obj to a Customer object

Ex2 will be graded from 0-100%.

50% 1,2 complete, 60%:1,2,3 complete, 70-85%:1,2,3,4 complete. 85%+ all parts complete, consideration of code quality including refactoring, validation etc.

*Guidance of creating a windows form application – refer to ‘GUI in VS’ under week2 and ‘Stack in a GUI’ under week4 in Moodle*