# **RecursiveExercise02 (binary search tree)**

In this exercise you should learn some basic about using recursion on a data structure that also has a recursive description:
In this case we will be using a binary search tree, when balanced it can contain a lot of elements and still be really fast for maintain and for searching and can also be fast for getting elements in order.
You will find similar recursive structure in xml, html, bill of materials and figures.

Concrete you will learn about

* How a primitive binary tree can be made with a self-referenced structure
* how to use recursion for traverse through a binary tree
* how to use recursion for searching after data in a binary tree

Example of organizing binary search tree – elements are called nodes. (More about concepts can be read in Danish here – sorry I haven’t got time for translation: <http://bjoerks.net/CSharp/noter/BinaereTraeer.htm>)

Principle is for all node is all node to the left less than or equal and all notes to the right greater.

For this exercise you will have at startup solution with some code already implemented:
- [RecursiveSolution02\_BinTree.zip](http://bjoerks.net/klasser/SK2016F_dm15abc/RecursiveSolution02_BinTree.zip)

**The next two exercises must be performed with a person from another class different from the one in the last exercise, so go find someone.**

## **Exercise – Write\_InOrder\_Recursiv:**

1. Start debugging through insertion of the data step by step to see how the test tree is build.
2. Now implement the method for traversing through the test tree and write all data in the tree in accending order (inorder) using recursion – data is just to be written at console directly from the method – don’t care about user output from model (**chief-secretary**).
3. Debug through your method to figure out problems if it don’t work as expected.
4. When done, your group find another group of 2 person and exchange your experience.

## **Exercise – Search\_Recursiv:**

1. Now implement the method for searching for data in the binary tree (**chief-secretary** – switch!).
2. Debug through your method to figure out problems if it don’t work as expected.
3. When done, your group find another group of 2 person and exchange your experience.