# Lab5 Assignment – Binary SearchTrees

Do these problems in order. While implementing/debugging, hard-code the program's input in your Python file. Once your code is working you can prompt the user for input.

**Problem 1: BST operations.**

In this problem you will implement the Ordered Dictionary ADT operations using Binary Search Trees. You will have to define a BinarySearchTree class, with the methods for **`search, minimum, maximum, successor, predecessor, insert, delete`**. You can use the class TreeNode and its helper methods discussed in the class/tutorial.

To test your code, you can create a BST in the main (or in a separate function) and run the operations on that tree.

To print a BST, you can define & use a method **`preorder-traverse`** that prints the pre-order traversal of the keys in the BST.

**Problem 2: Parse Trees.**

a) Write a program to build a parse tree from an fully parenthesised expression.
b) Write functions **`printPostfix`** and **`printPrefix`** that take as input an expression parse tree and prints the postfix and prefix representation of a parse tree.
c) Write a function **`evaluateTree`** to evaluate the value of a expression parse tree.

**Problem 3.*** Print the *level order traversal* of a binary tree. Given a binary tree, you should first print the root (level 0), then its children (level 1), then their children (level 2),..and so on. Within a given level, the nodes should be printed from left to right.

**Problem 4*. W**rite a program to construct and count the number of unique Binary Search Trees with keys 1,2,...n. The keys of the nodes are distinct so each BST has exactly n nodes. Print the pre-order traversal of each unique tree.

For e.g. if n=2

2 unique trees. Pre-order traversals:

```
1 2
2 1
```

if n=3:

5 unique trees. Pre-order traversals:

```
1 2 3
1 3 2
2 1 3
3 1 2
3 2 1
```

* Additional Exercises.