**Pracise 4 Report**

**1. Is the tree that is created from the node files (dict \* .dat) complete or almost complete? Justify your answer.**

A complete tree is one in which every node other than the leaves has two children. A complete binary tree is one in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible. To fulfil this aspect, the tree must have an even number of nodes, and all these files contain a multiple of ten number of nodes. We can conclude that the dict \*.dat are almost complete trees.

**2. a) What is the relationship between the "shape" of a tree and its traversing modes?**

In the pre-order mode of traversing, the root is the first element to be read, followed by the left child and finishing in the right one.

In the In-order mode of traversing, we first visit the left child, followed by the root, and finishing in the right child.

And in the Post-order mode of traversing, we start in the left child, then the right child, and finish at the root.

We can easily observe all these modes of traversing a tree vary on when the root is being visited.

**b) Can you tell if a binary search tree is well constructed according to how it is traversed?**

By looking at the last node of every leave we can deduce if the tree is well constructed. Traversing a tree in Post-Order provide us that possibility, looking first at the last node of the most left leave, and from there to the most right one.

**3. Compare and describe the differences between the trees generated by the executables p4\_e3 with the last argument B or N (number of nodes, depth, routes, etc.).**

­­While executing p4\_e3 with any dict\*.dat we can easily observe that with the same number of nodes, the depth is always lower with the B argument (balanced). This option follows a route in order to maintain the depth at its lowest. Therefore, the non-balanced option requires more time to search for a specific node.