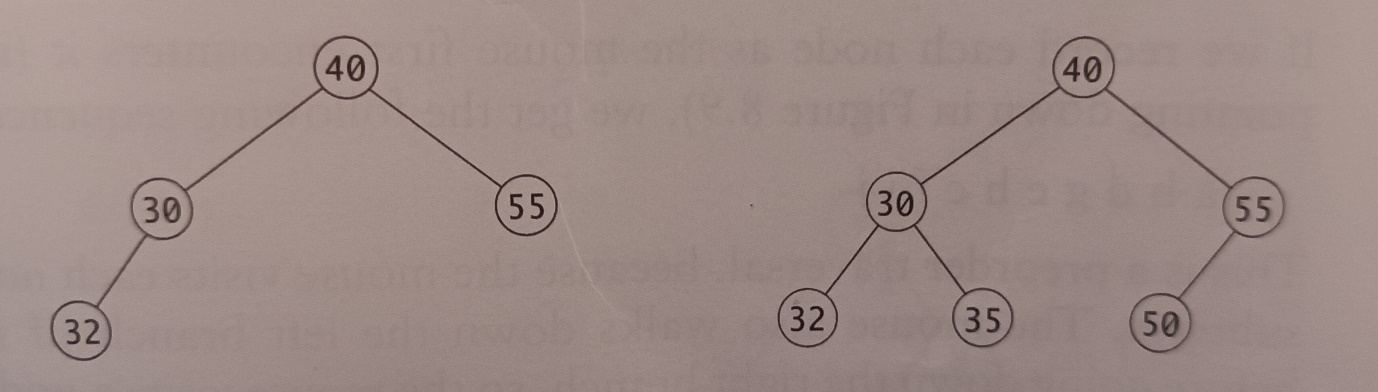
1. Produce binary search trees for the following data sets. Indicate whether the trees created are full or complete.
   1. 18,11,35,9,42,33,14,6,10,27,34
   2. Jack, Peter, Alex, Michael, Stewart, Bill, Simon, Charlie
   3. 50,42,45,55,53,70,36,40,90,71,33
2. Draw binary expression trees for the following infix expressions. Your trees should enforce the Java rules for operator evaluation( BODMAS).
3. a \* b
4. a \* b + c
5. x / y + a - b \*c
6. For each tree below answer the questions:

What is its height? Is it a full tree? Is it a complete tree? Is it a binary search tree? If not, make it a binary search tree.



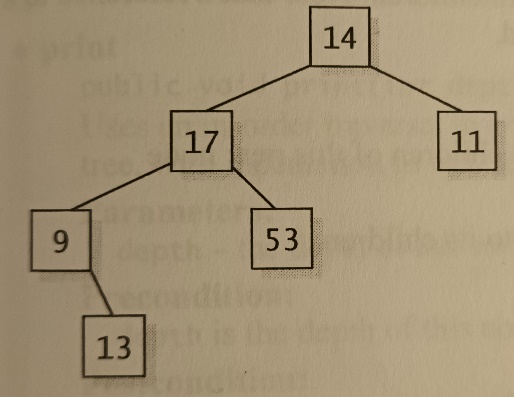
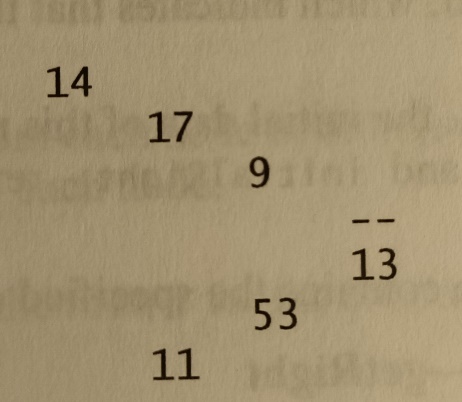
**Figure 1**

1. For the trees shown above provide the inorder, preorder and postorder traversal.
2. Being able to display a binary tree can be very useful. A pre-order traversal that prints each node with an indentation to indicate its depth can be implemented. A dash is printed if there is a child that has no sibling. For the following binary tree in Figure 2 the display in Figure 3 can be created.

Is this a binary search tree? \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Is this tree full? \_\_\_\_\_\_\_\_\_\_\_\_

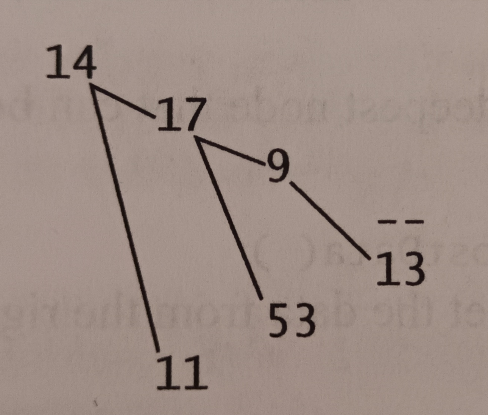
Is this tree complete? \_\_\_\_\_\_\_\_\_\_\_\_

**Figure 2 Figure 3**

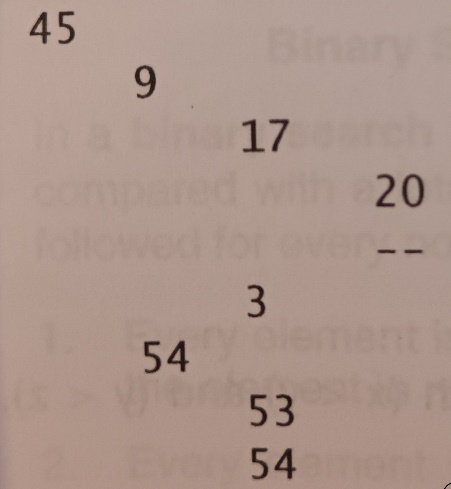
* 14 is the root node at depth 0 so it is not indented at all.
* Nodes 17 and 11 are at depth 1 so are indented by one space.
* Nodes 9 and 53 are at depth 2 so are indented by two spaces.
* Finally mode 13 is at depth 3 so is indented 3 spaces.

We can draw lines to connect a node to its children.



**Figure 4**

For the output below draw lines to connect nodes to their children. Then draw the binary tree.



Is this a binary search tree? \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Is this tree full? \_\_\_\_\_\_\_\_\_\_\_\_

Is this tree complete? \_\_\_\_\_\_\_\_\_\_\_\_