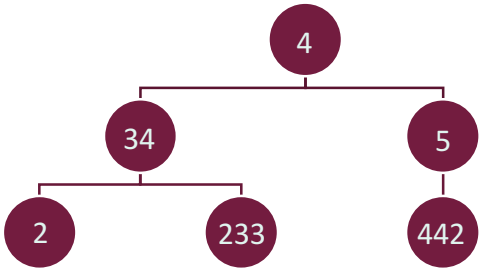
Supplementary Sheet 2

Trees and Heaps

Heaps are the one of the coolest data structures ever. They are a type of binary trees. A tree is a data structure where there exists a root node. The root node will have a value; it could be an integer e.g. 5329 or a string. The root can have other nodes as children. And these nodes can also have children, etc.



In the tree, to the left, 4 is the root node, and it has two children 34 and 5. The node labelled 5, has a child, with value 442. The node labelled 34, has two children 2 and 223.

Conversely, the parent node of 442 is 5, and the parent node of 5 is 4.

A binary tree is where each node has at most two children. The first child is called the left child and the second child is called the right child. The tree to the right constitutes as a binary tree.

A heap is a data structure based on a binary tree. The additional criterion that makes a binary tree a heap, is that it satisfies the "heap property". There are two variants of heaps, and consequently, two variants of the heap property. In a max heap, the parent node of a given node must be greater than or equal to that given node. E.g. the above binary tree cannot be a max heap, as there many violations of this rule, e.g. 34 is greater than 4. A min heap is the opposite, the parent node of any given node must be less than or equal to that given node. The above binary tree could be a min heap, if the node labelled 2 had a value greater or equal to 34.

Adding Items to A Heap

When adding items to a heap, you always fill the previous row, before adding a new row. You also always fill from left to right. You start by adding the item at the next position from the left at the bottom of the heap. If you are writing a max heap, the heap will then compare the new node with its parent to see if the node is less than the new node. If this is the case, it will swap the nodes. It keeps on doing this, till either a node is found that is greater than the new node, or the new node becomes the root. E.g. with the max heap below: