AQA Computer Science A-Level

4.3.2 Tree-traversal

Advanced Notes

Specification:

4.3.2.1 Simple tree-traversal algorithms

Be able to trace the tree-traversal algorithms:

- pre-order
- post-order
- in-order.

Be able to describe uses of tree-traversal algorithms. Pre-Order: copying a tree. In-Order: binary search tree, outputting the contents of a binary search tree in ascending order. Post-Order: Infix to RPN (Reverse Polish Notation) conversions, producing a postfix expression from an expression tree, emptying a tree.

Tree-Traversal

Synoptic Link

A **tree** is a **connected acyclic graph**.

Trees are covered in **Trees** under **Fundamentals of Data Structures**.

Algorithm

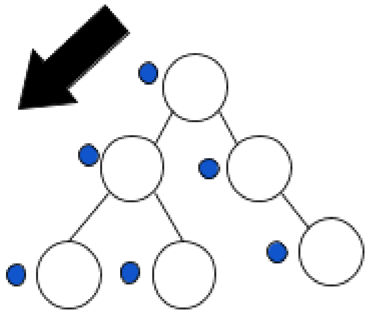
An algorithm is a set of instructions which completes a task in a finite time and always terminates.

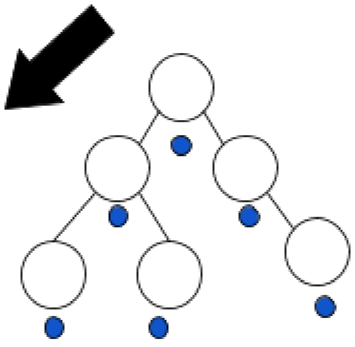
Tree-traversal is the process of visiting/updating/outputting each node in a tree - it is a form of algorithm. Unlike a graph-traversal, tree-traversals are unique to trees and must start at the root. From the root, they travel left, down the tree. There are three types of tree-traversals; pre-order, in-order and post-order. Pre-order and post-order tree-traversal can be performed on any tree including binary trees but an in-order traversal is only well defined for binary trees.

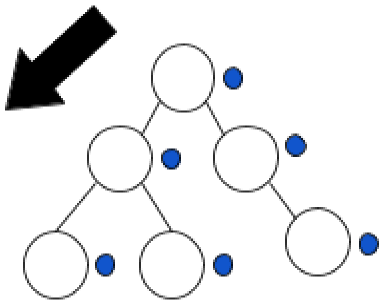
Synoptic Link

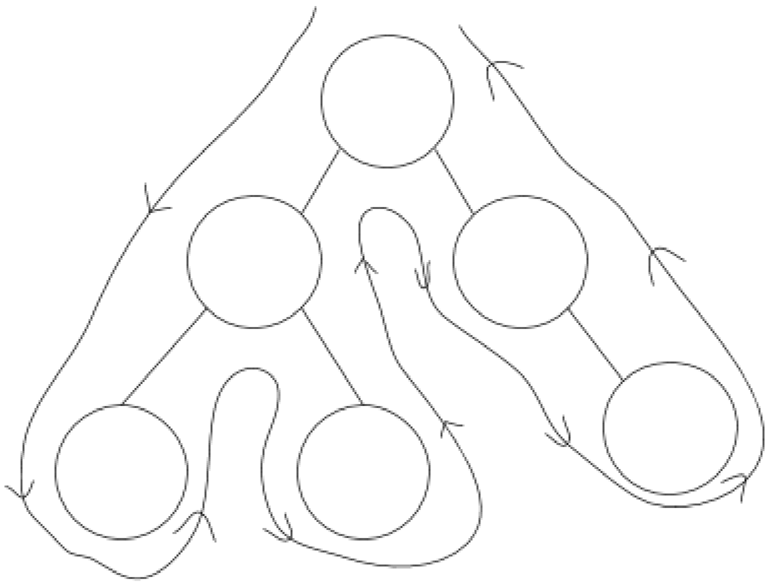
Graph-traversal is the process of **visiting** each **vertex** in a **graph**. **Tree-traversals** are a form of **depth-first** traversal.

Graph-traversal is covered in **Graph-traversal** under **Fundamentals of Algorithms.**


Pre-order


In-order


Post-order


The journey around a tree always occurs like this.

Pre-Order Traversal

Pre-order traversal is used for copying a tree. It can be performed on any tree.

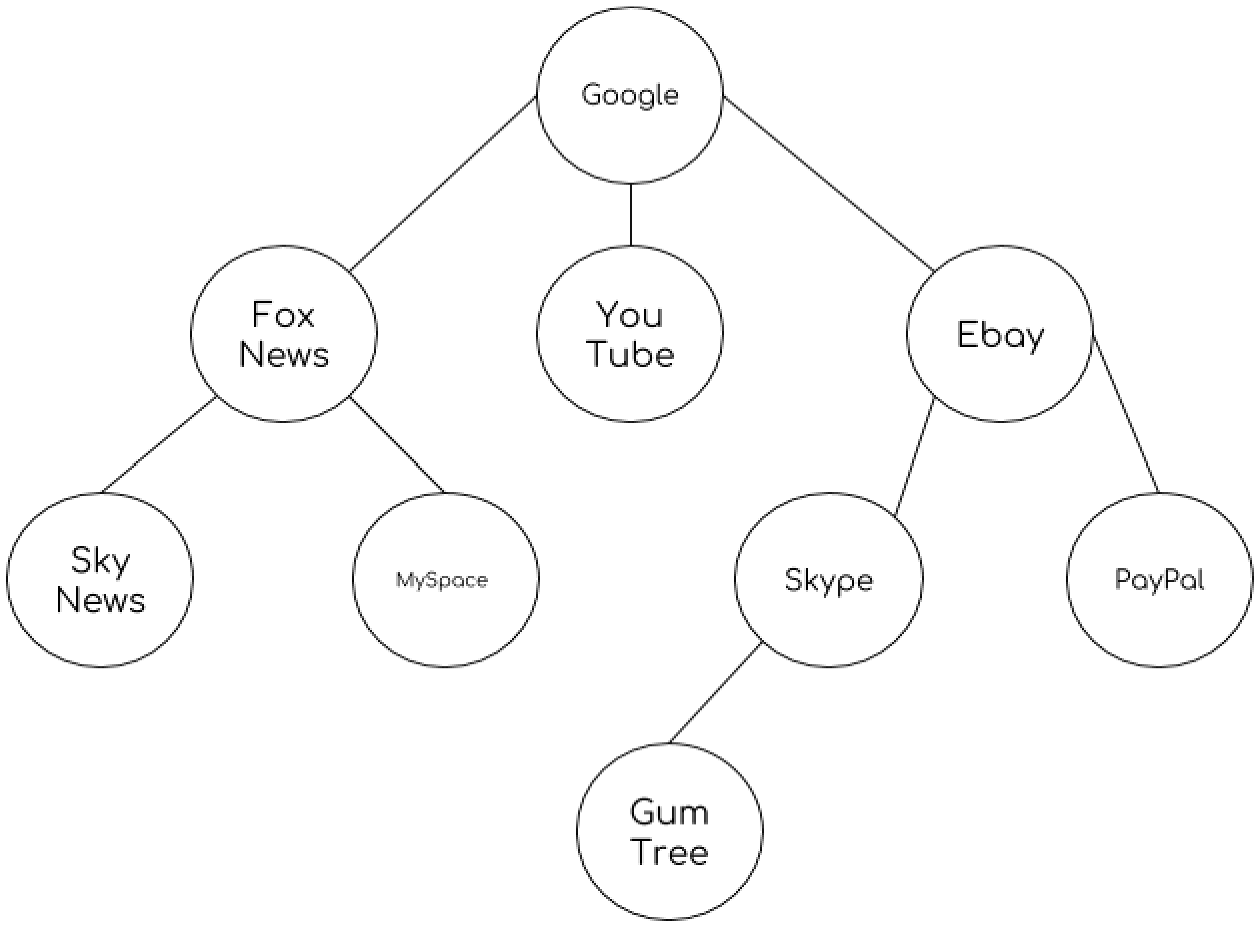
Example:

Here is a tree which shows the hierarchical relationships of companies owned by Google. This is **not** a binary tree because Google has more than two children. In this example, Google is the root.

Synoptic Link

A **binary tree** is a **rooted tree** where each node has at most **two** children. The **root node** has **no parent**.

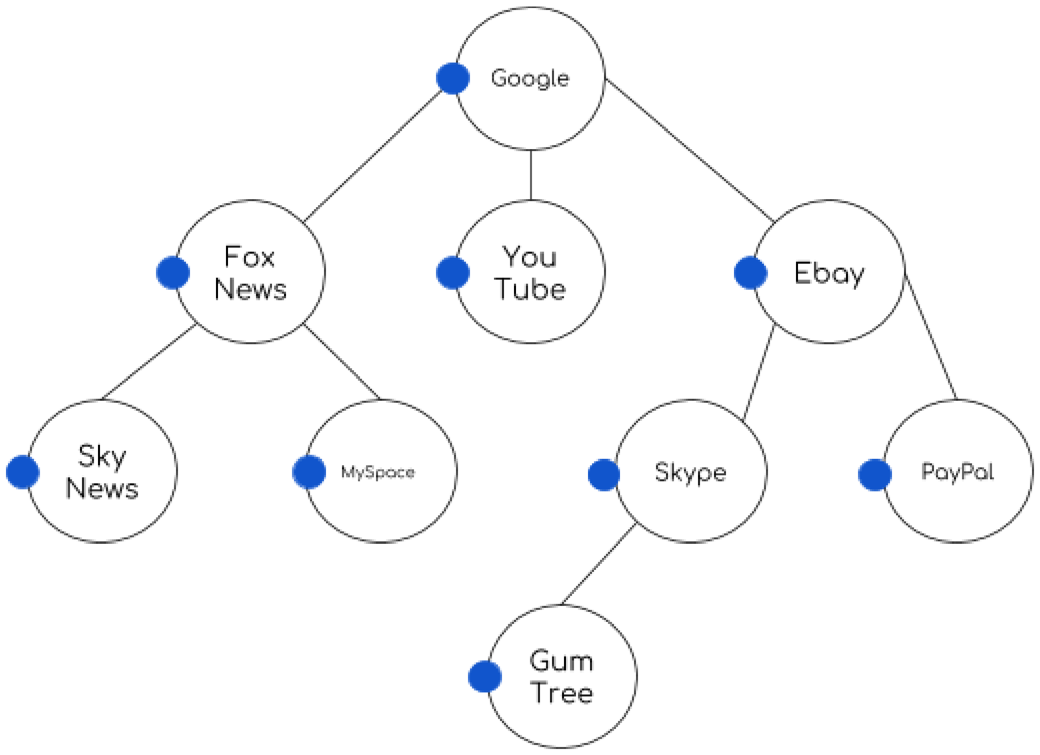
Binary trees are covered in **Trees** under **Fundamentals of Data Structures**.



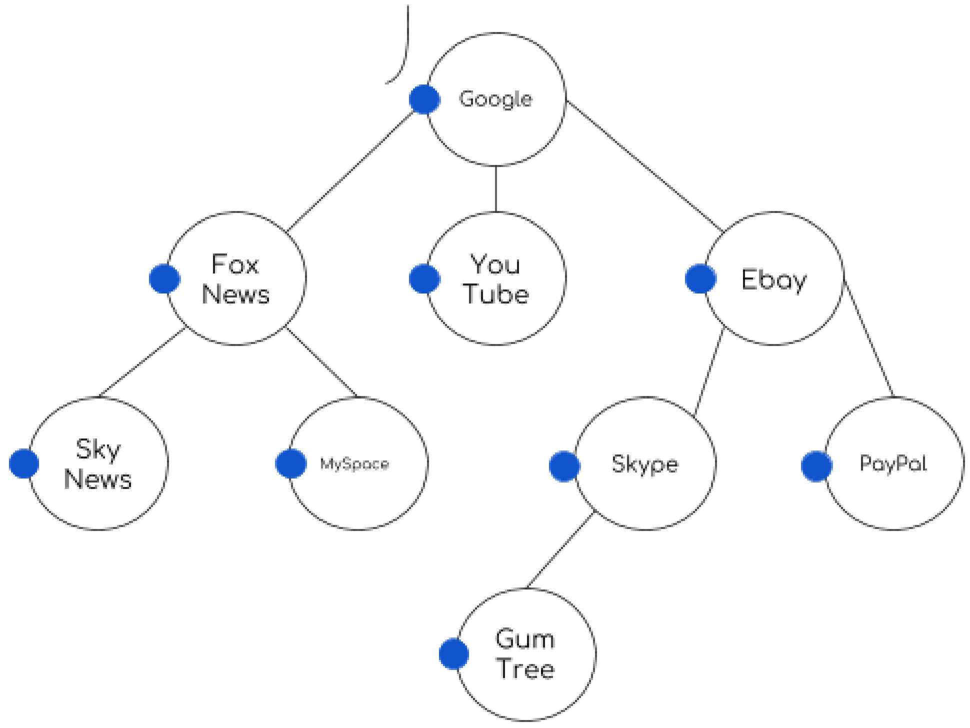
Leaf

A vertex/node can also be known as a leaf when it is on a tree.

When performing a pre-order traversal the first step is to mark the left hand side of each leaf.

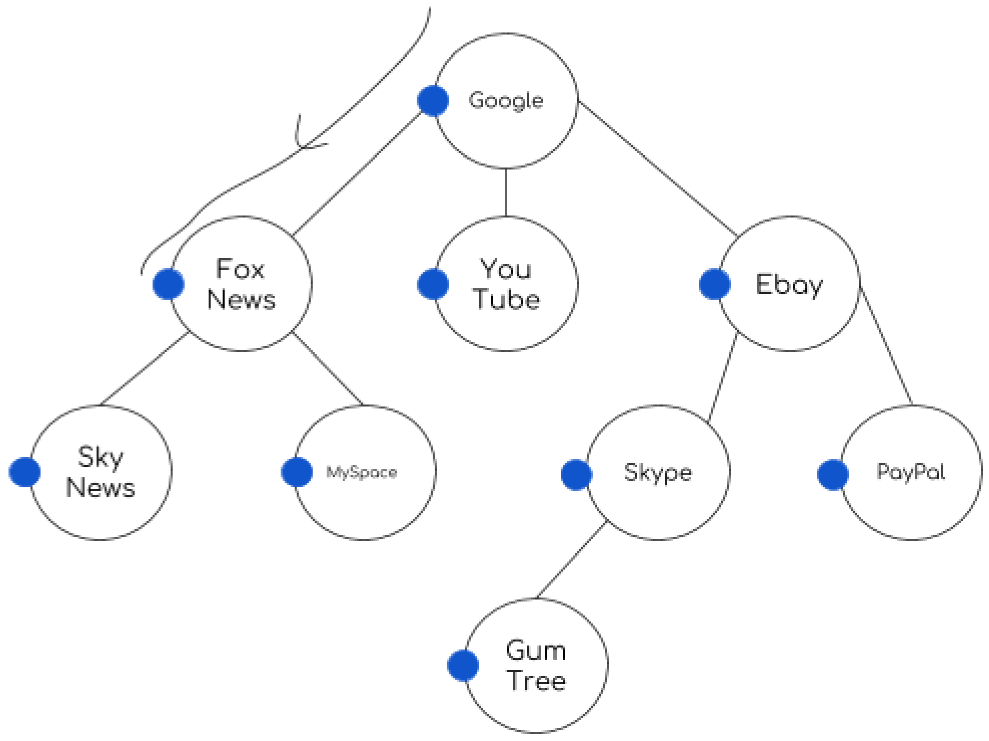


The traversal starts from the left and works down the tree. Whenever a blue spot is passed, the information on the node is outputted.



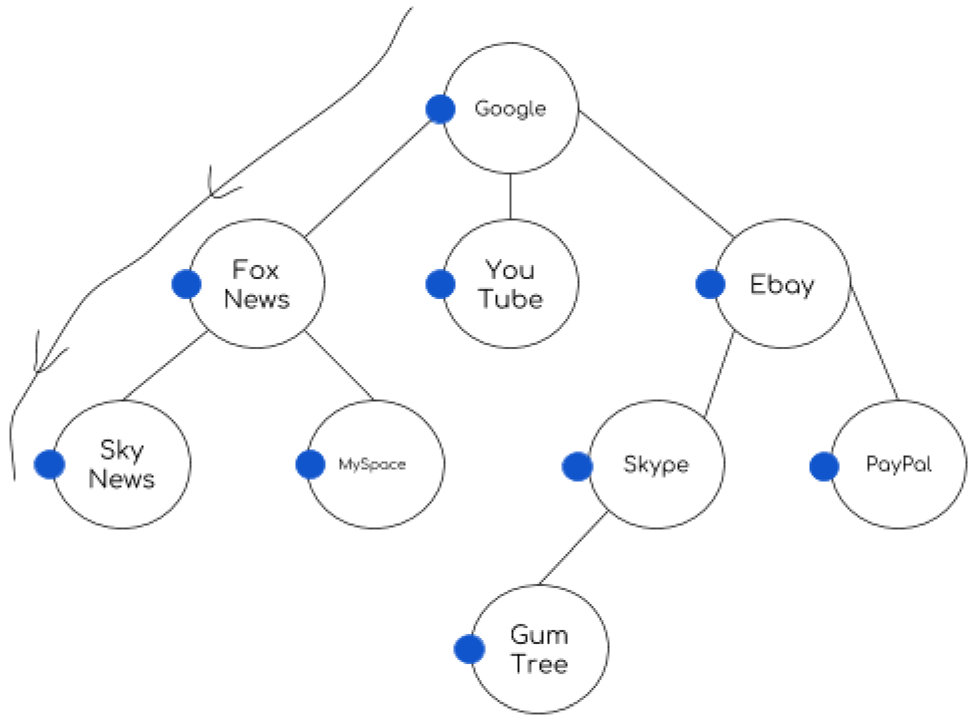
Output: Google

The blue spot on Google has been passed, so Google is outputted.



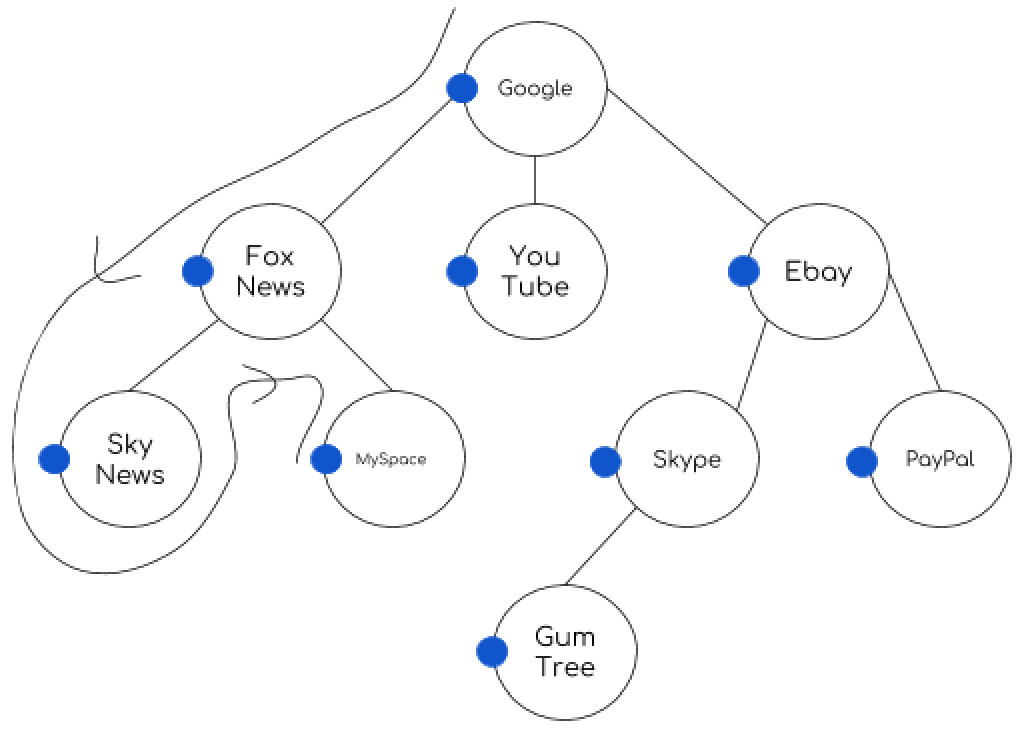
Output: Google, Fox News

Fox News has been passed, and outputted.



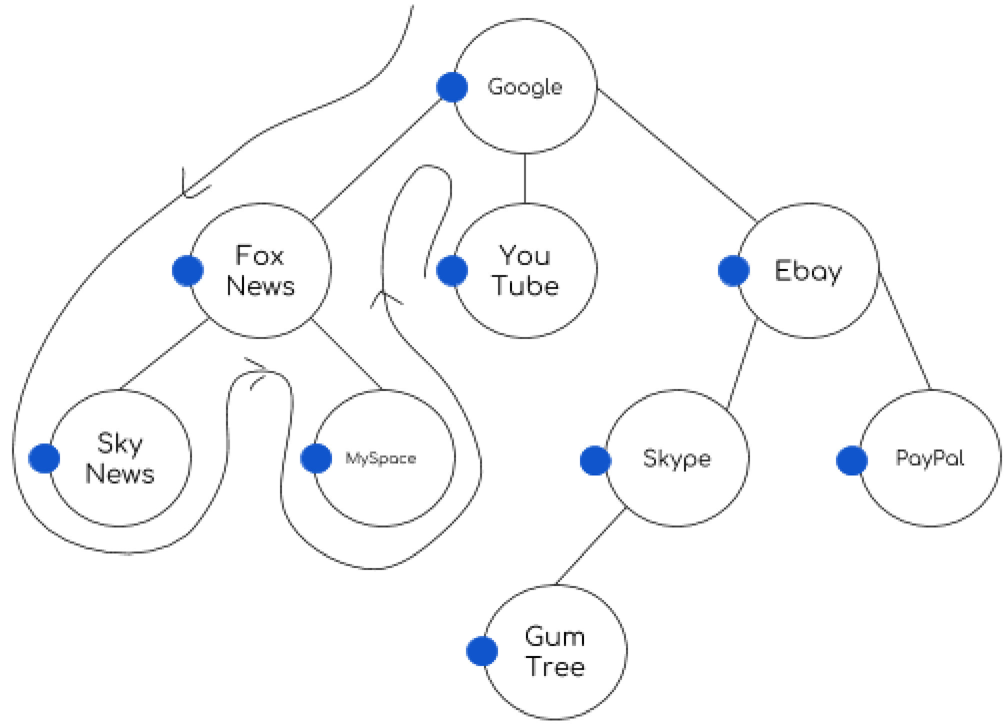
Output: Google, Fox News, Sky News

Sky News has been passed and outputted.



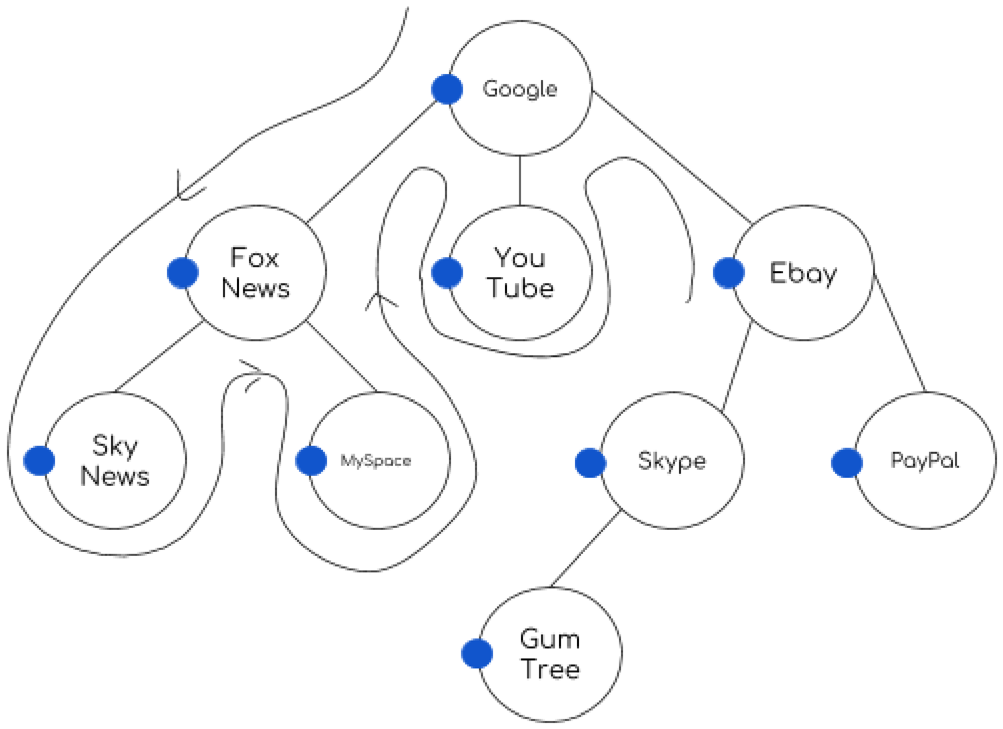
Output: Google, Fox News, Sky News, MySpace

MySpace has been passed and outputted.



Output: Google, Fox News, Sky News, MySpace, YouTube

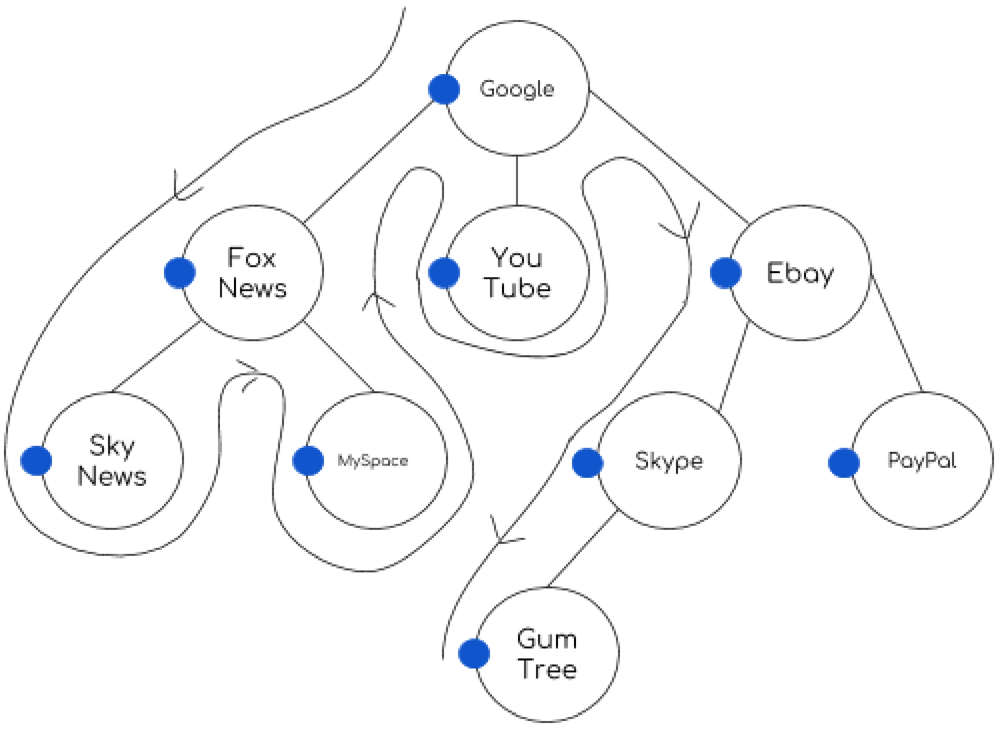
Youtube has been passed and outputted.



Output: Google, Fox News, Sky News, MySpace, YouTube, Ebay

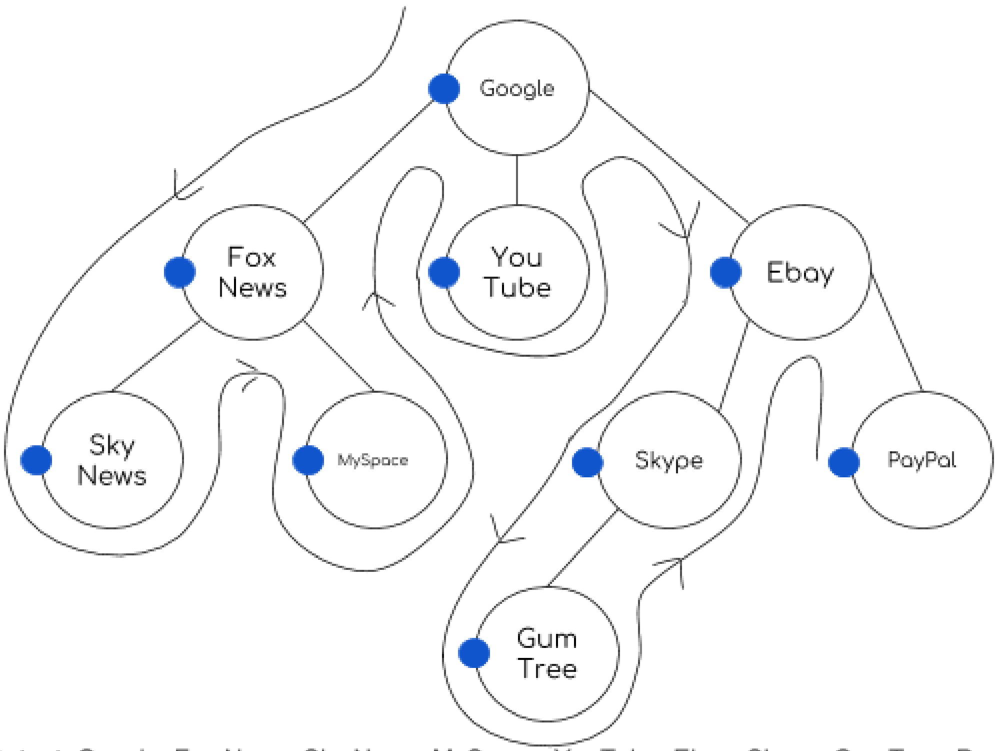
Ebay has been passed and outputted.

Skype has been passed and outputted.



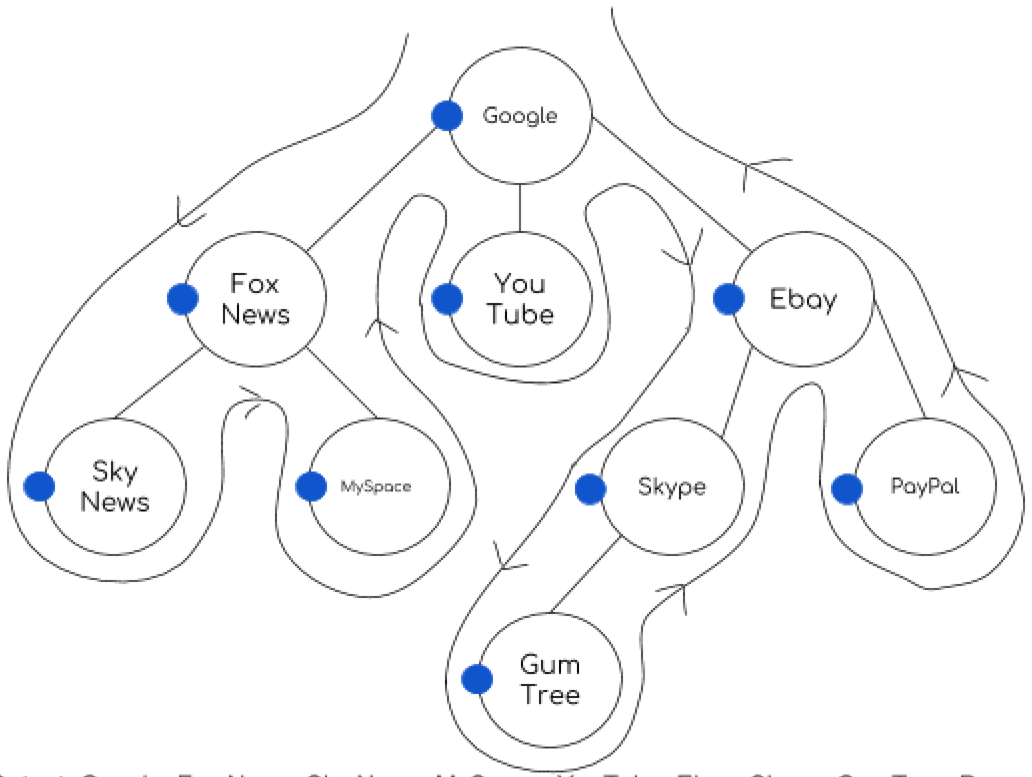
Output: Google, Fox News, Sky News, MySpace, YouTube, Ebay, Skype, GumTree

GumTree has been passed and outputted.



Output: Google, Fox News, Sky News, MySpace, YouTube, Ebay, Skype, GumTree, Paypal

PayPal has been passed and outputted.



Output: Google, Fox News, Sky News, MySpace, YouTube, Ebay, Skype, GumTree, Paypal

Note

Traversals are algorithms and algorithms always terminate.

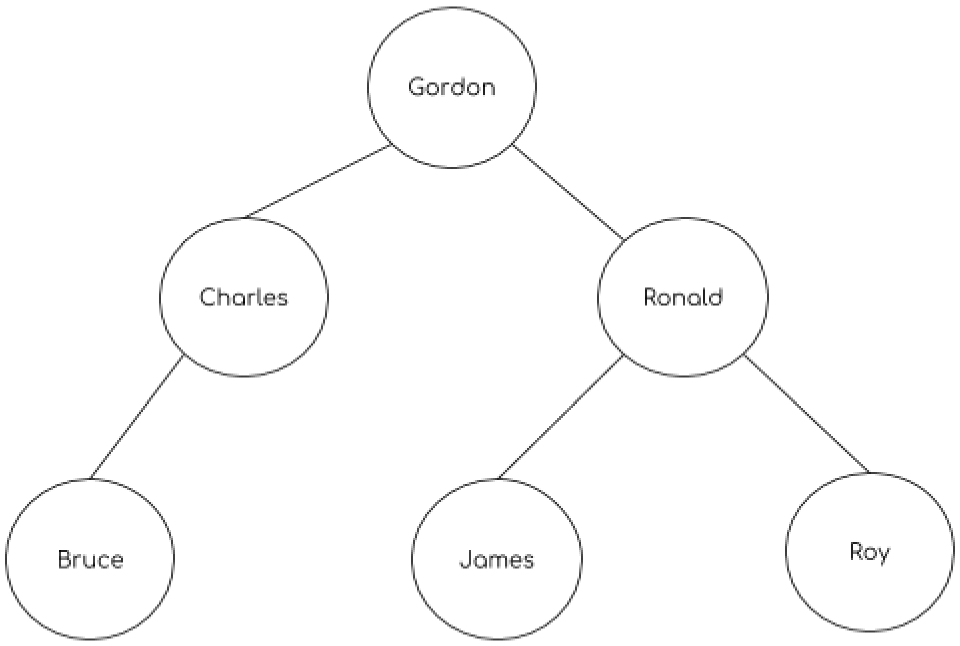
The traversal has completed. If all the nodes and edges were copied, an exact duplicate node could be made from a preorder traversal.

In-Order Traversal

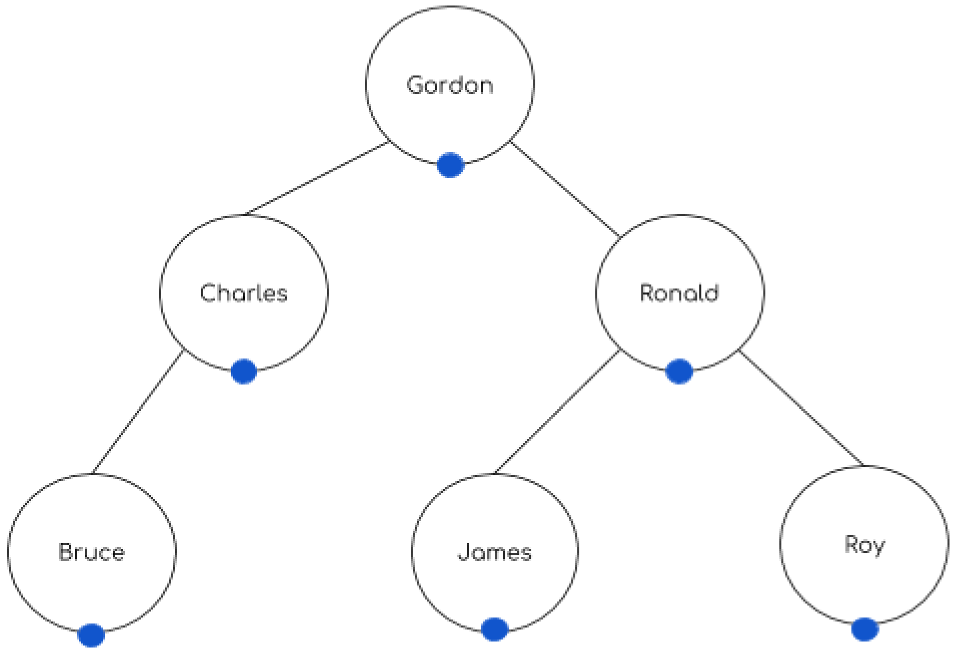
In-order traversal is useful for a binary search tree and because it will output the contents of a binary search tree in ascending order. It can only be performed on binary trees.

Example:

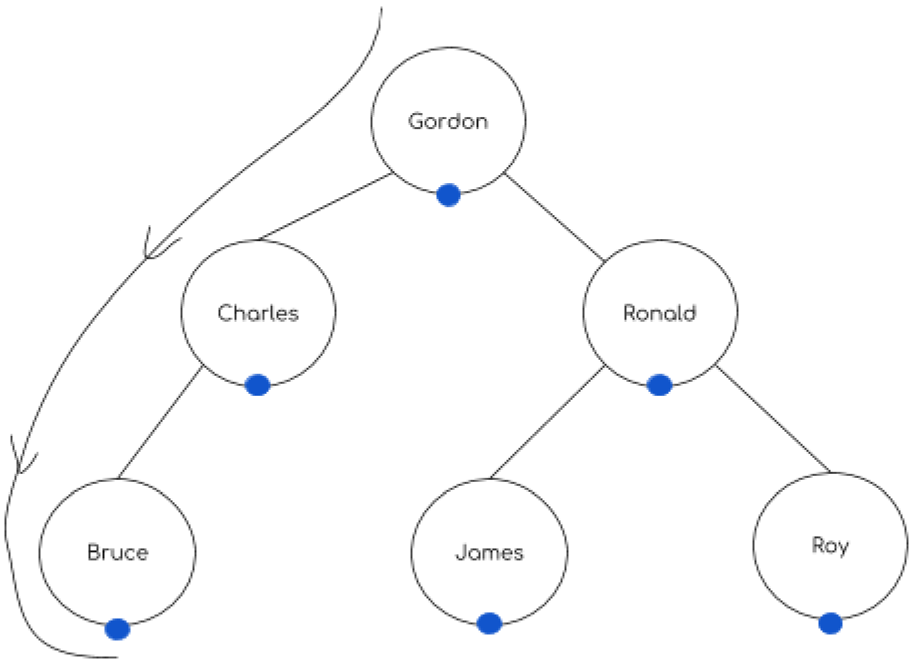
Here is a binary tree.



The first step in an in-order traversal is to mark the bottom of the children and parents.

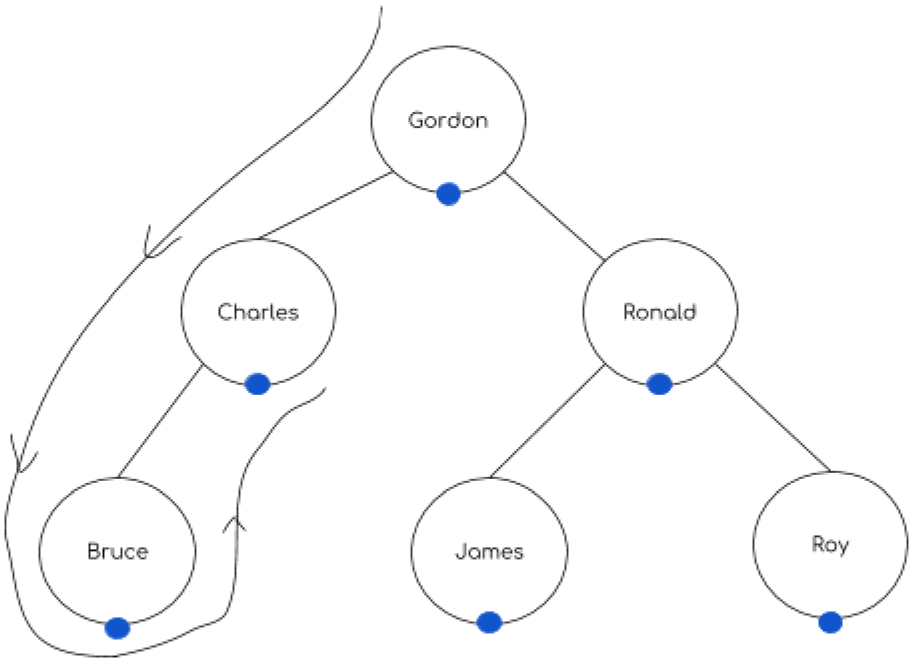


The traversal starts from the left and works around the tree. When a blue spot is passed, the node is outputted.



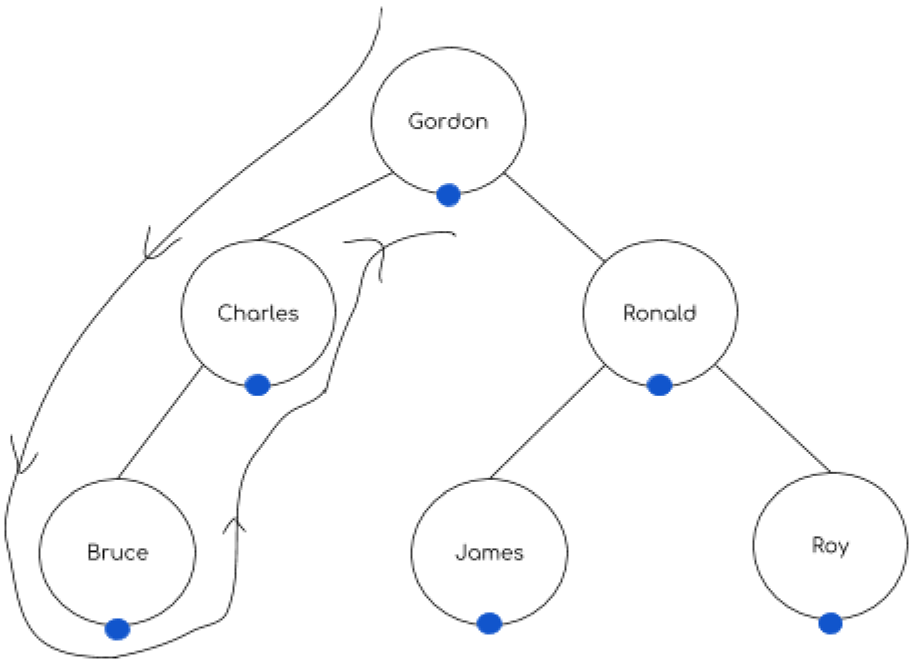
Output: Bruce

Bruce has been passed and outputted.



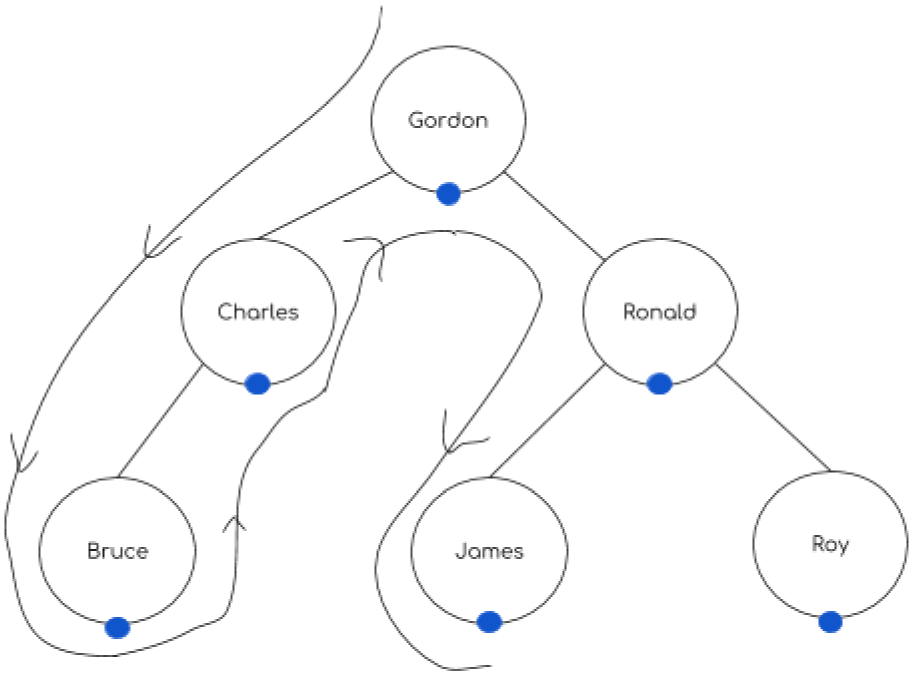
Output: Bruce, Charles

Charles has been passed and outputted.



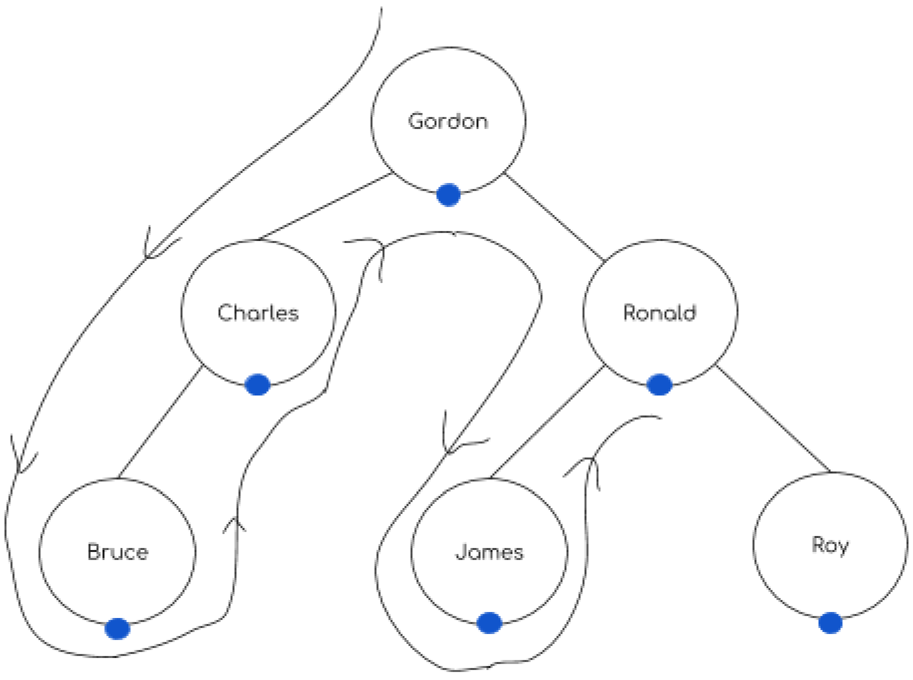
Output: Bruce, Charles, Gordon

Gordon has been passed and outputted.



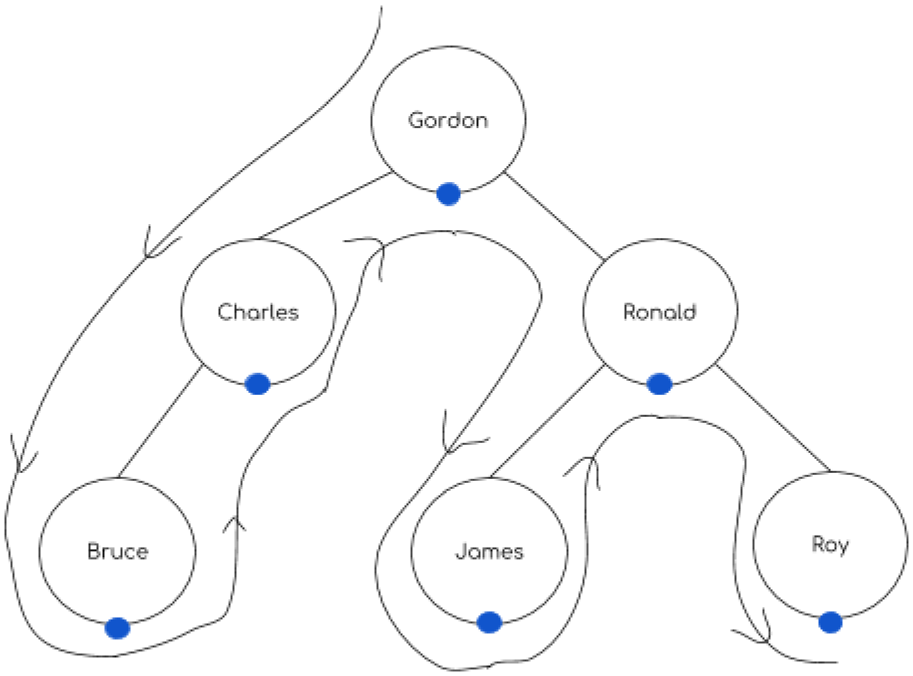
Output: Bruce, Charles, Gordon, James

James has been passed and outputted.



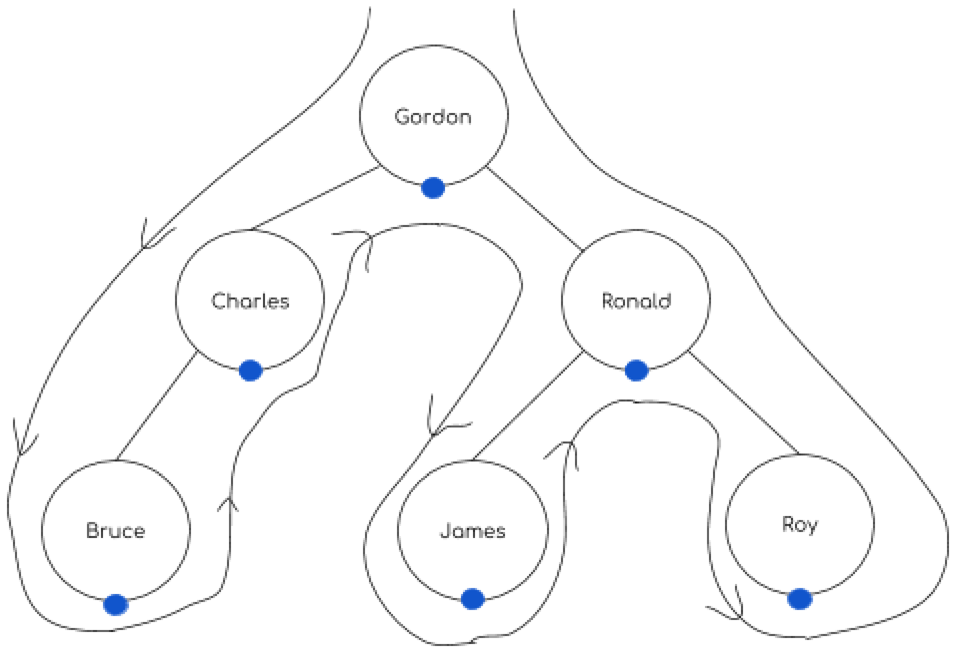
Output: Bruce, Charles, Gordon, James, Ronald

Ronald has been passed and outputted.



Output: Bruce, Charles, Gordon, James, Ronald, Roy

Roy has been passed and outputted.



Output: Bruce, Charles, Gordon, James, Ronald, Roy

The traversal has completed.

The above example used a special kind of binary tree called a binary search tree. As you can observe from the diagram, the nodes have been outputted in alphabetical order, which is a function of the in-order traversal.

Synoptic Link

A **binary search tree** is an **ordered binary tree**.

Binary search trees are covered in **Trees** under **Fundamentals of Data Structures**.

Synoptic Link

Reverse Polish Notation is a postfix way of writing expressions. RPN eliminates confusion over the order of execution and the need for brackets.

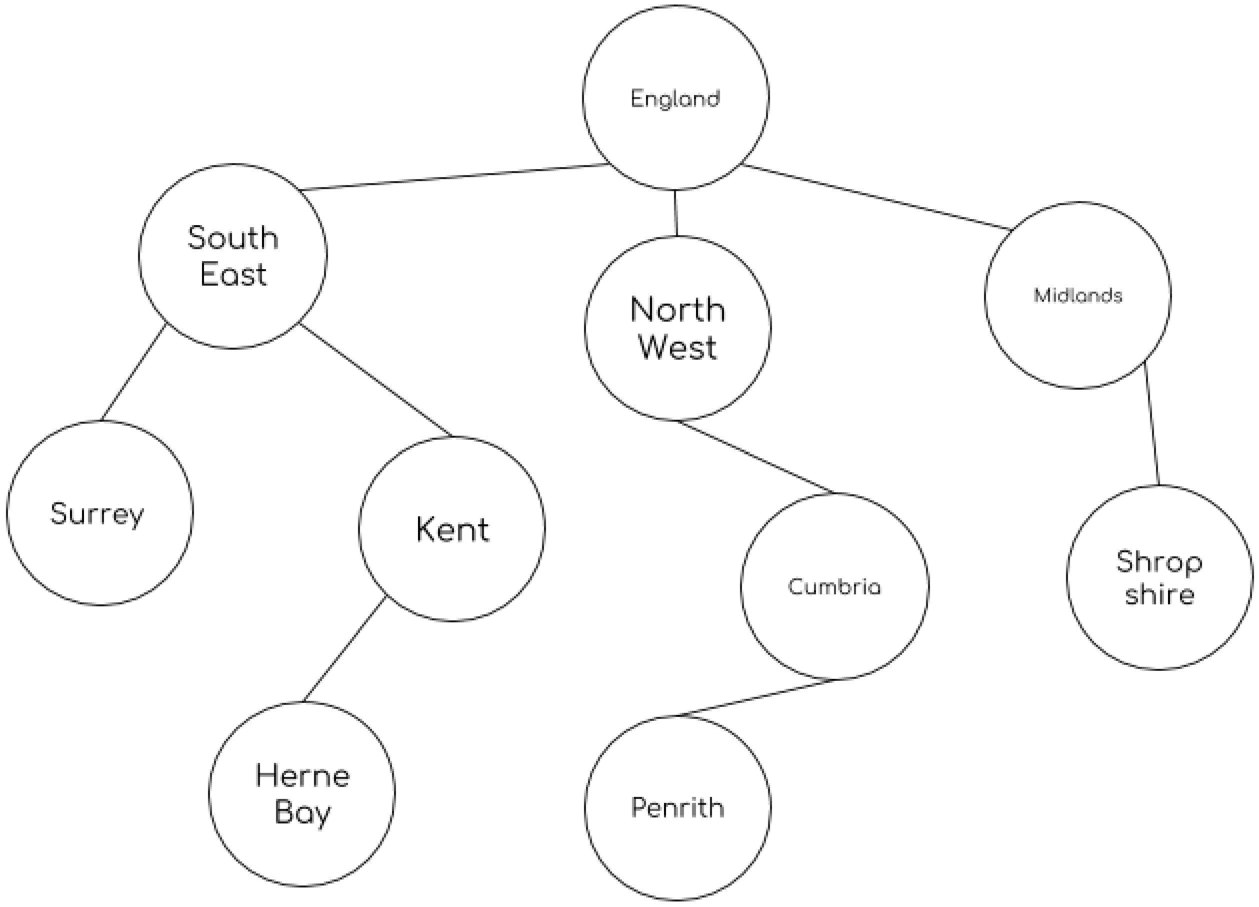
Reverse Polish Notation is covered in **Reverse Polish** under **Fundamentals of Algorithms**.

Post-Order Traversal

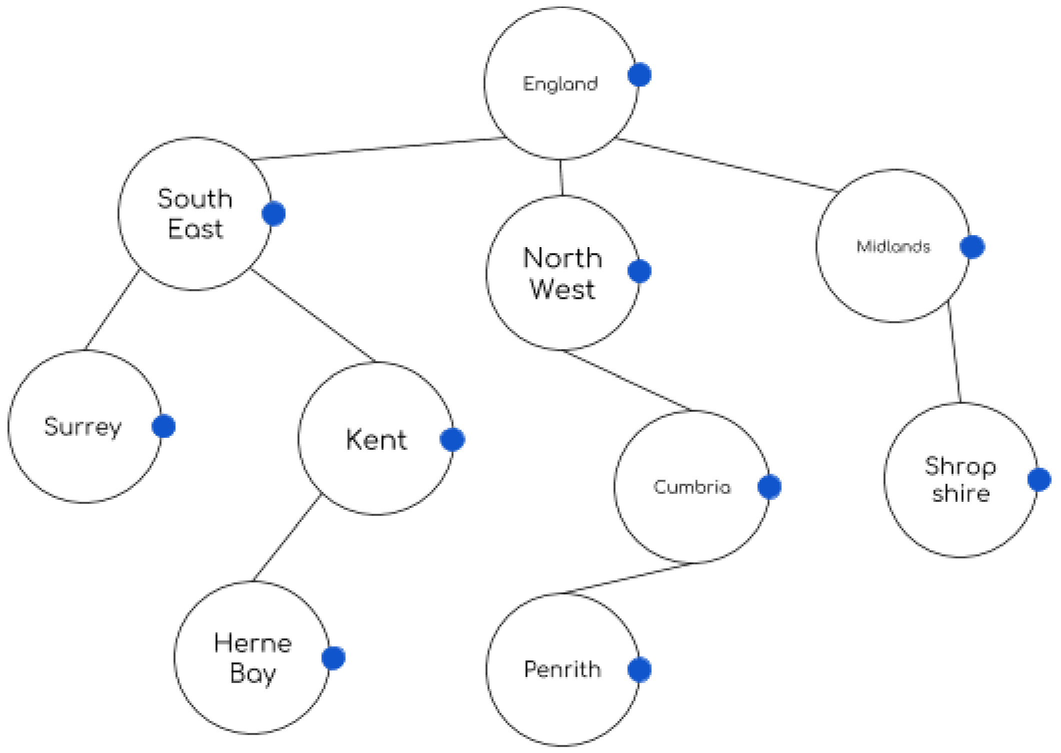
Post-order traversals can be performed on any tree. They are useful for Infix to RPN (Reverse Polish Notation) conversions, producing a postfix expression from an expression tree and emptying a tree.

<u>Example:</u>

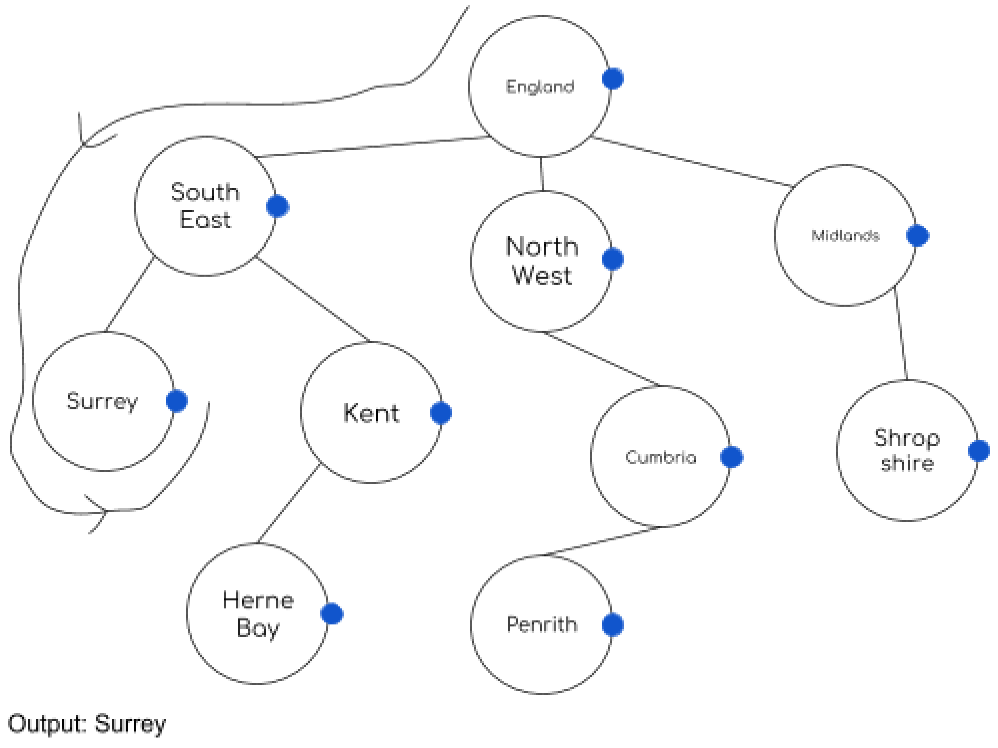
Here is a tree for some of the constituent parts of England.



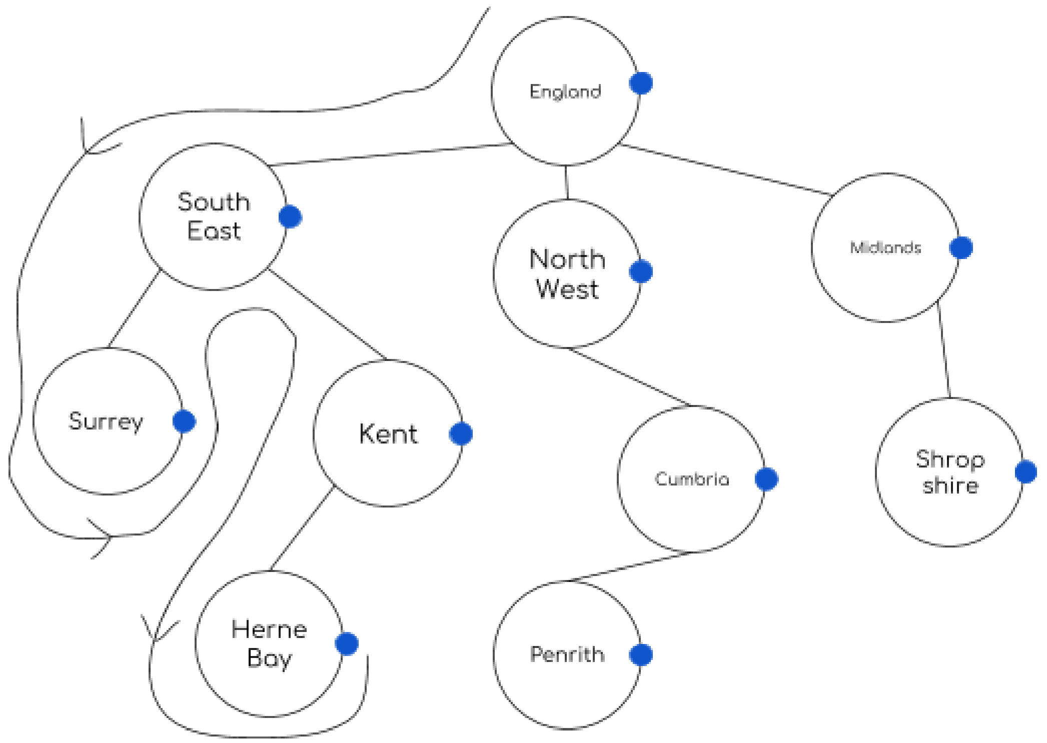
The first step in post-order traversal is to mark the right hand side of the nodes.



The traversal starts from the left and works its way around the nodes. As the traversal passes the blue dots the node is outputted.

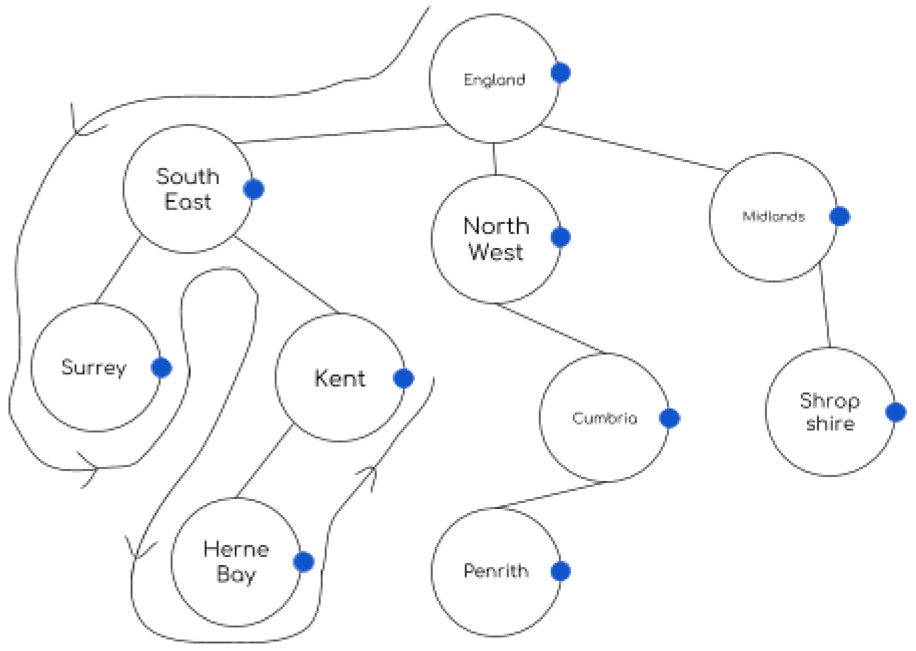


Surrey has been passed and outputted.



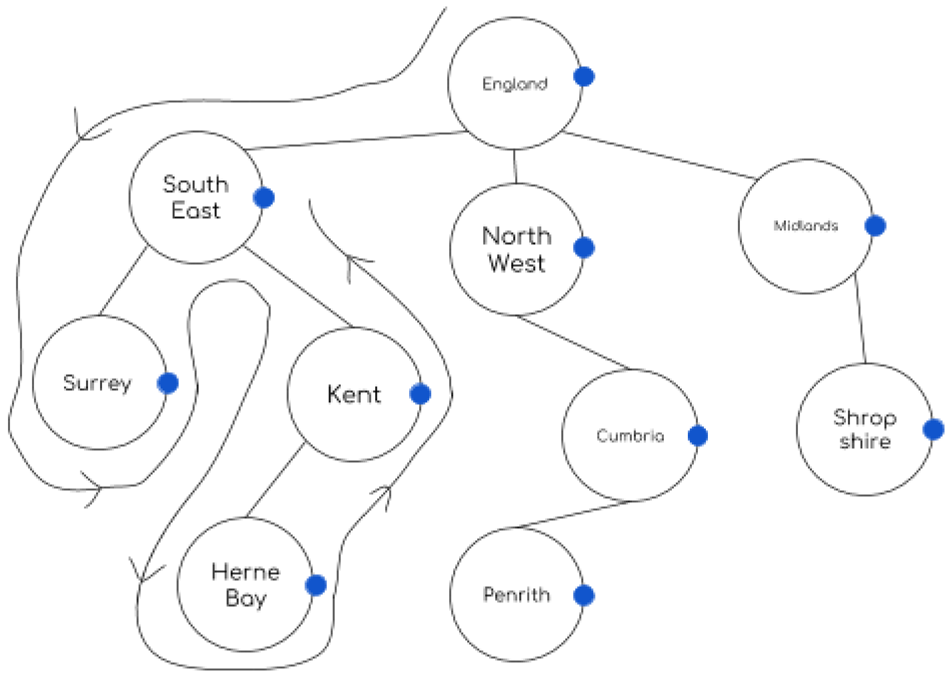
Output: Surrey, Herne Bay

Herne Bay has been passed and outputted.



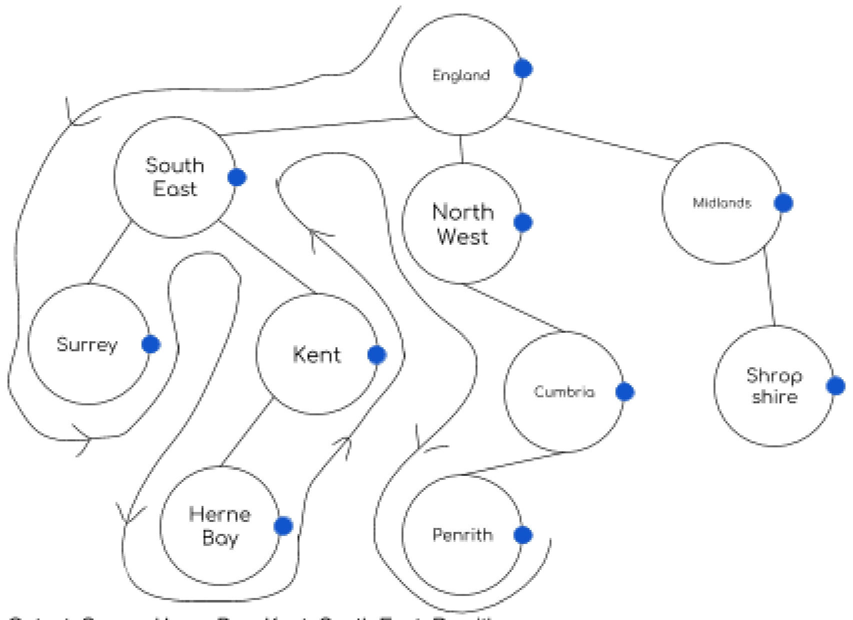
Output: Surrey, Herne Bay, Kent

Kent has been passed and outputted.



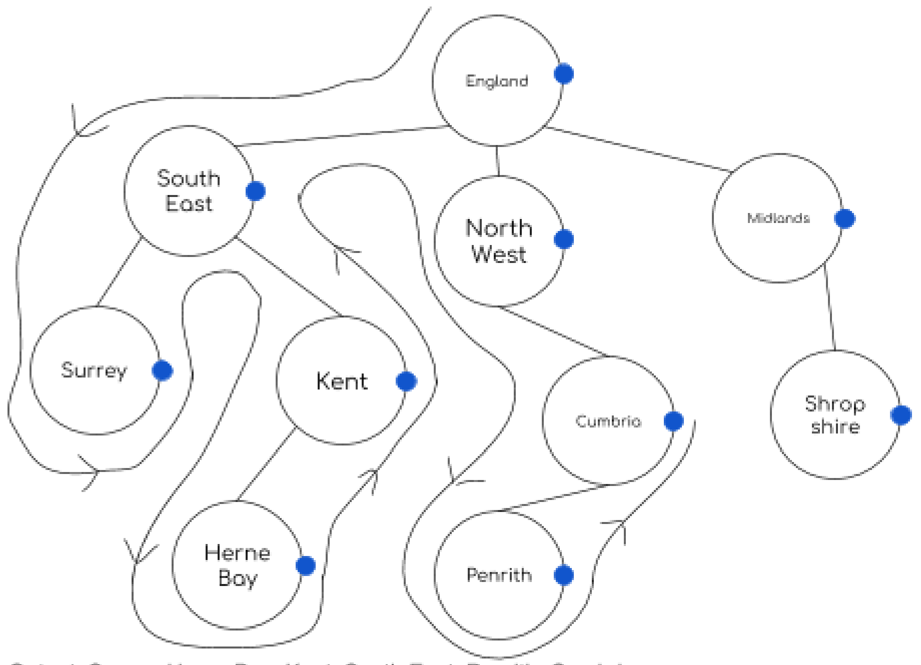
Output: Surrey, Herne Bay, Kent, South East

South East has been passed and outputted.



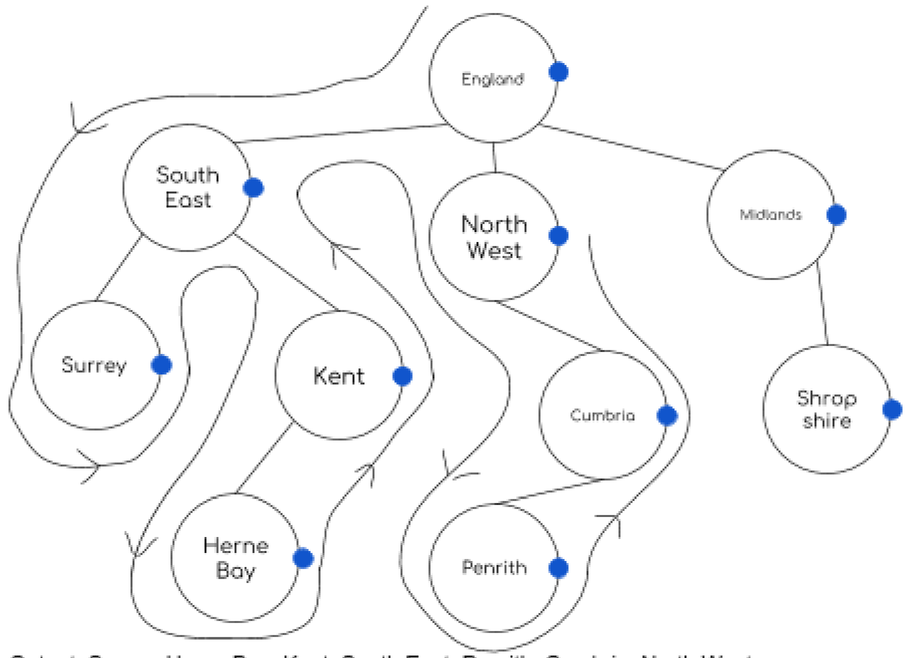
Output: Surrey, Herne Bay, Kent, South East, Penrith

Penrith has been passed and outputted.



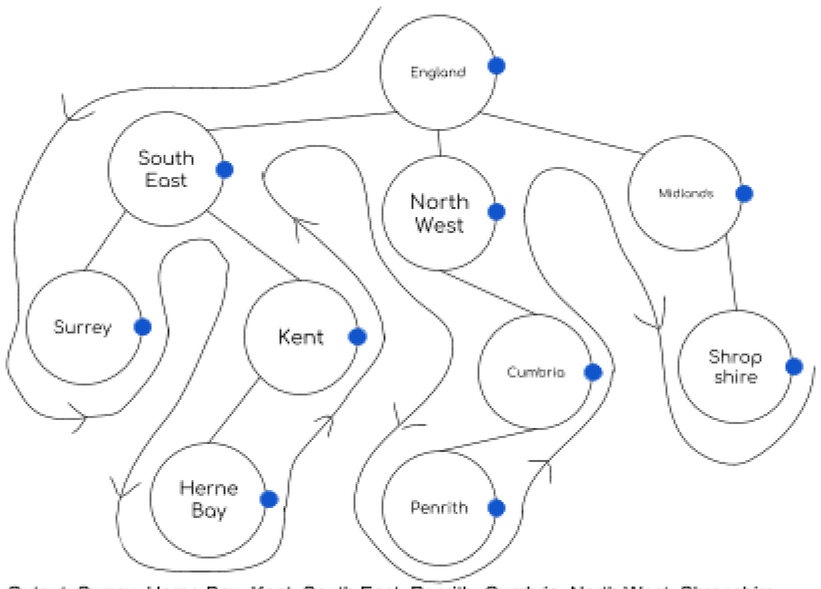
Output: Surrey, Herne Bay, Kent, South East, Penrith, Cumbria

Cumbria has been passed and outputted.



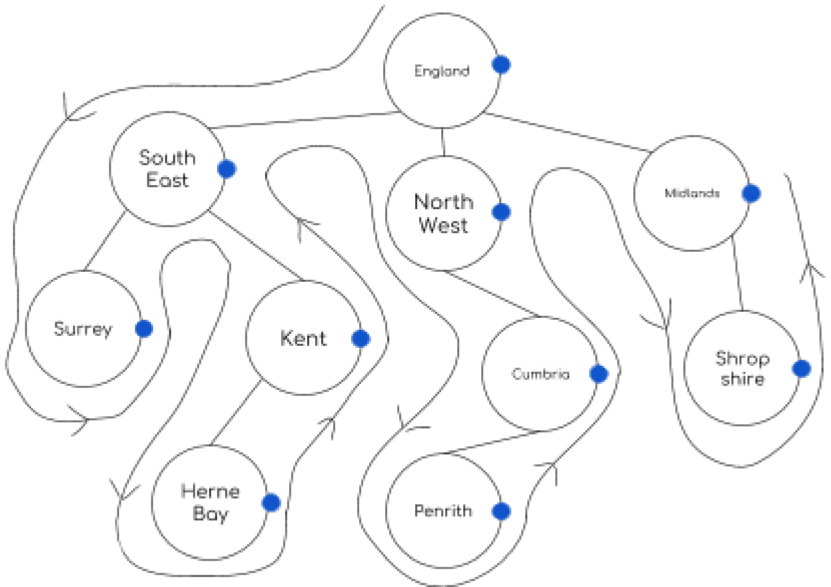
Output: Surrey, Herne Bay, Kent, South East, Penrith, Cumbria, North West

North West has been passed and outputted.



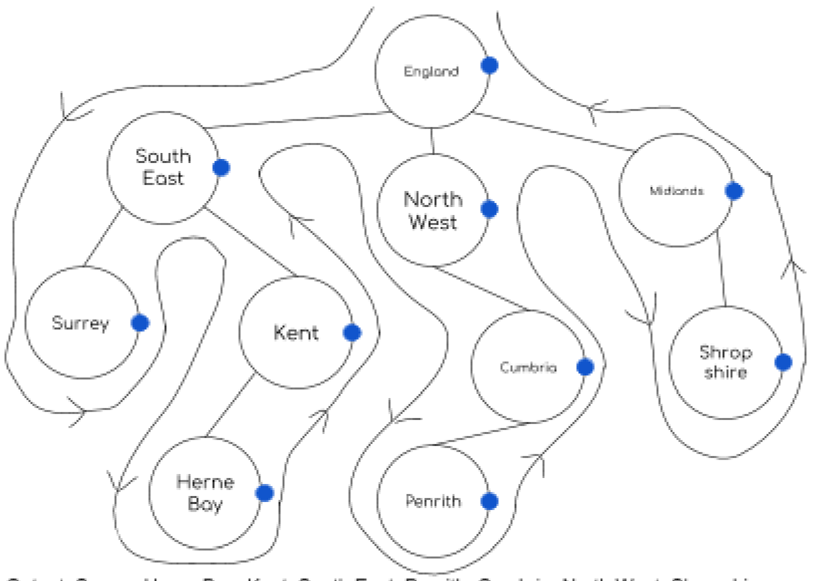
Output: Surrey, Herne Bay, Kent, South East, Penrith, Cumbria, North West, Shropshire

Shropshire has been passed and outputted.



Output: Surrey, Herne Bay, Kent, South East, Penrith, Cumbria, North West, Shropshire, Midlands

Midlands has been passed and outputted.



Output: Surrey, Herne Bay, Kent, South East, Penrith, Cumbria, North West, Shropshire, Midlands. England

England has been passed and outputted. The traversal has finished.

Infix to Postfix

As mentioned above, postorder traversal can be used to make infix to RPN conversions and make postfix expressions from expression trees. Technically, these two uses are the same. The difference is that Infix to RPN would involve the making of and then traversing of a binary tree, whereas in the latter it can be assumed the expression tree has already been formed.

Expression Trees

Trees wherein the nodes contain either opcode or an operand. Expressions can be derived from their traversal.

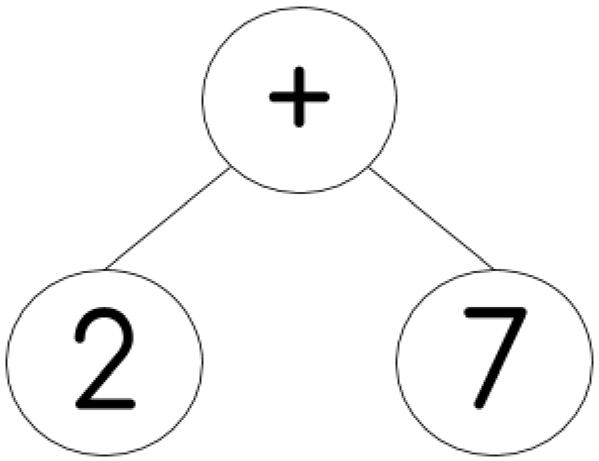
Synoptic Link

Opcode is an **operator**, **operand** is the **data** which is applied to the operator. E.g. in the expression 5 - 2 the - is the opcode and 5 and 2 are the operand.

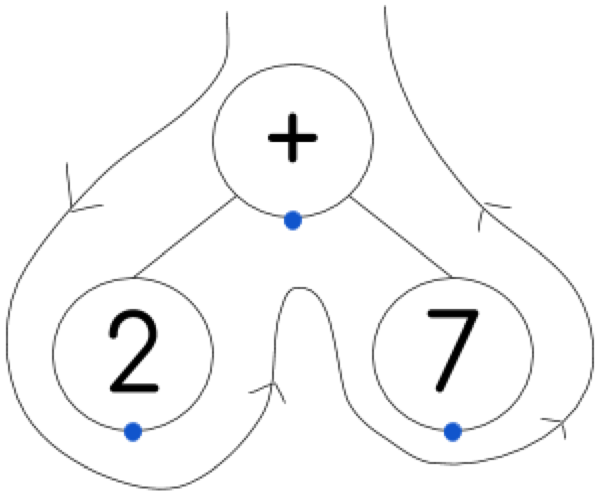
Opcode and operands are covered in **Structure and role of the processor and its components** under **Fundamentals of Computer Organisation and Architecture**

Example 1:

The infix expression 2 + 7 is given. It is needed to be converted into its postfix equivalent. 2 and 7 are the operand, and + is the opcode. The first step is to put the expression into an expression tree. The parent should be the opcode, and its two children are the operands.

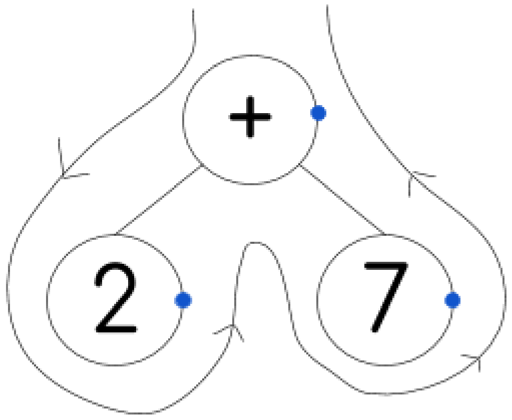


By performing an inorder traversal of this tree, you would get the infix expression.



Output: 2 + 7

Postorder traversal of the tree would produce the postfix expression 2 7 +.



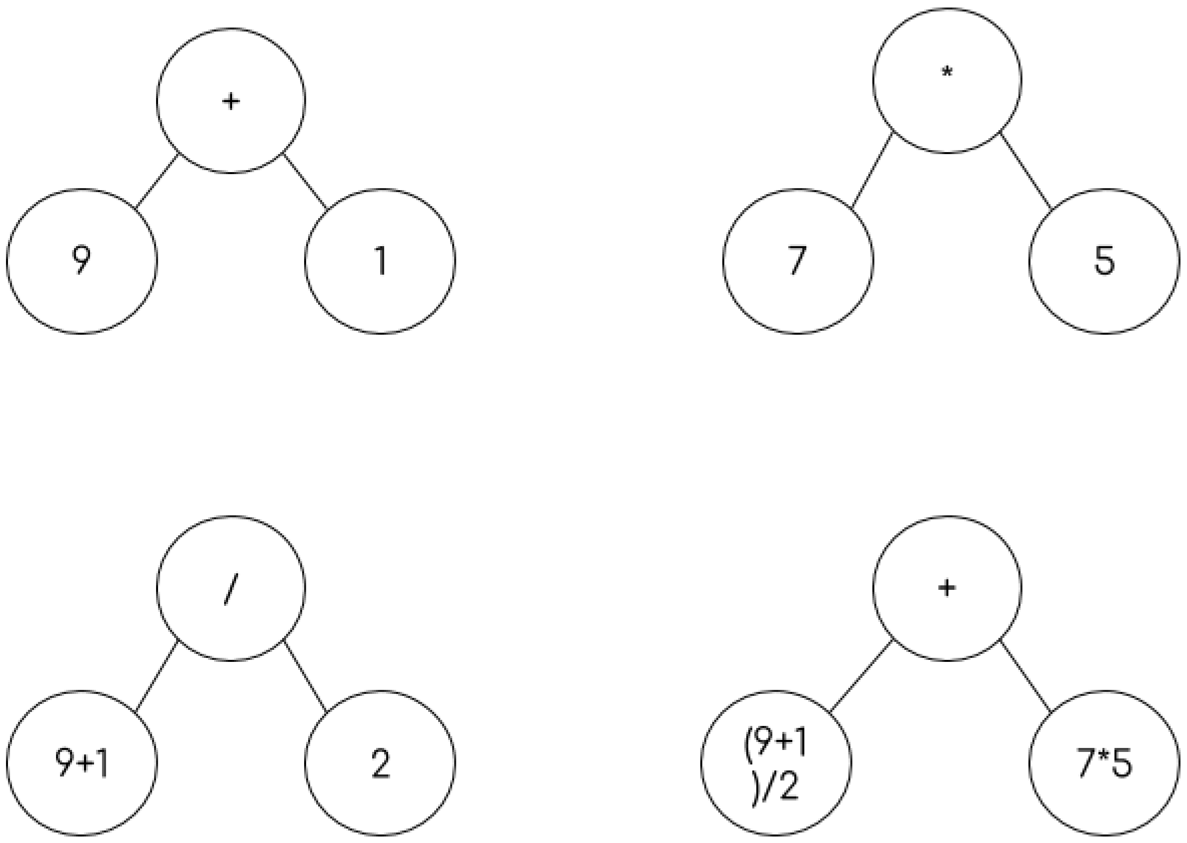
Output: 2 7 +

Example 2:

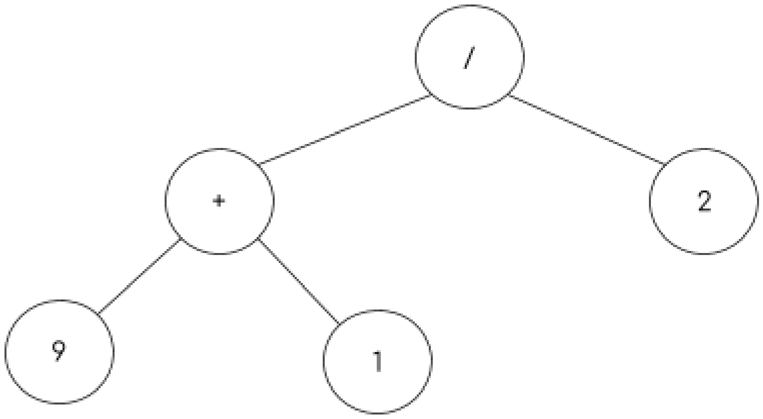
The expression (9 + 1)/2 + (7 x 5) is to be turned into its postfix equivalent. There are 4 operators, each of which will form its own subtree with its operand. Here are the sub trees - the opcode is the parent and the operands are the children.

Subtree

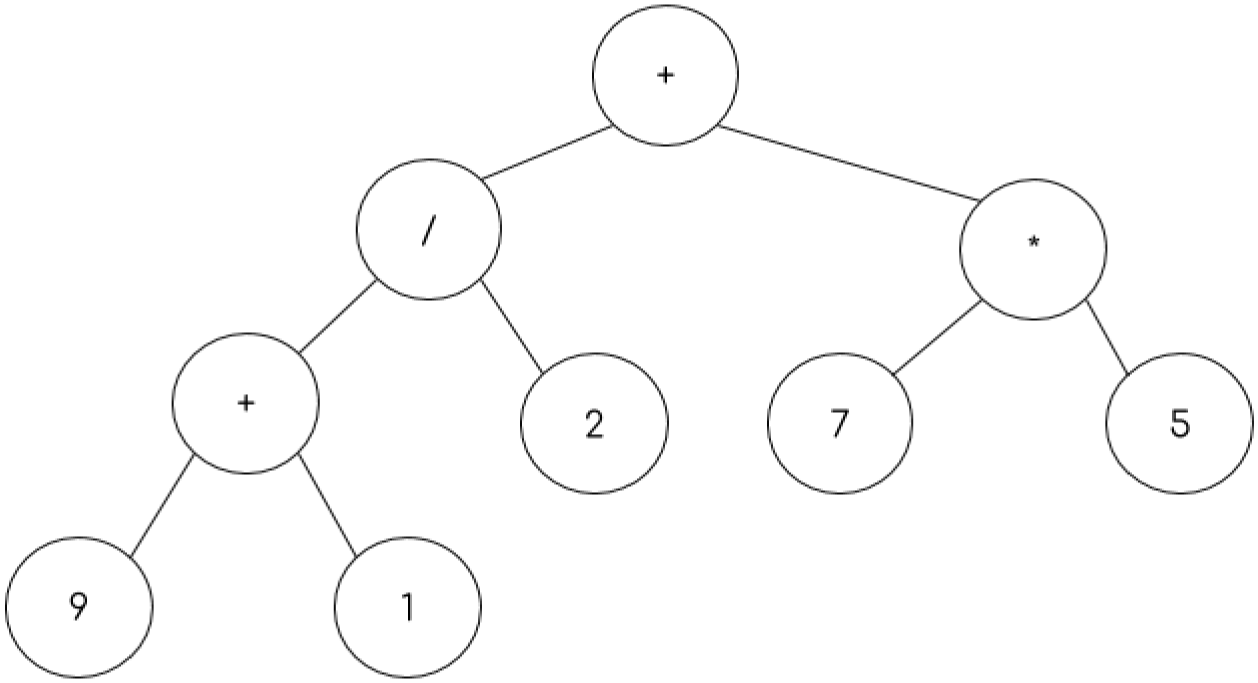
A tree which is descended from a node belonging to a larger tree.



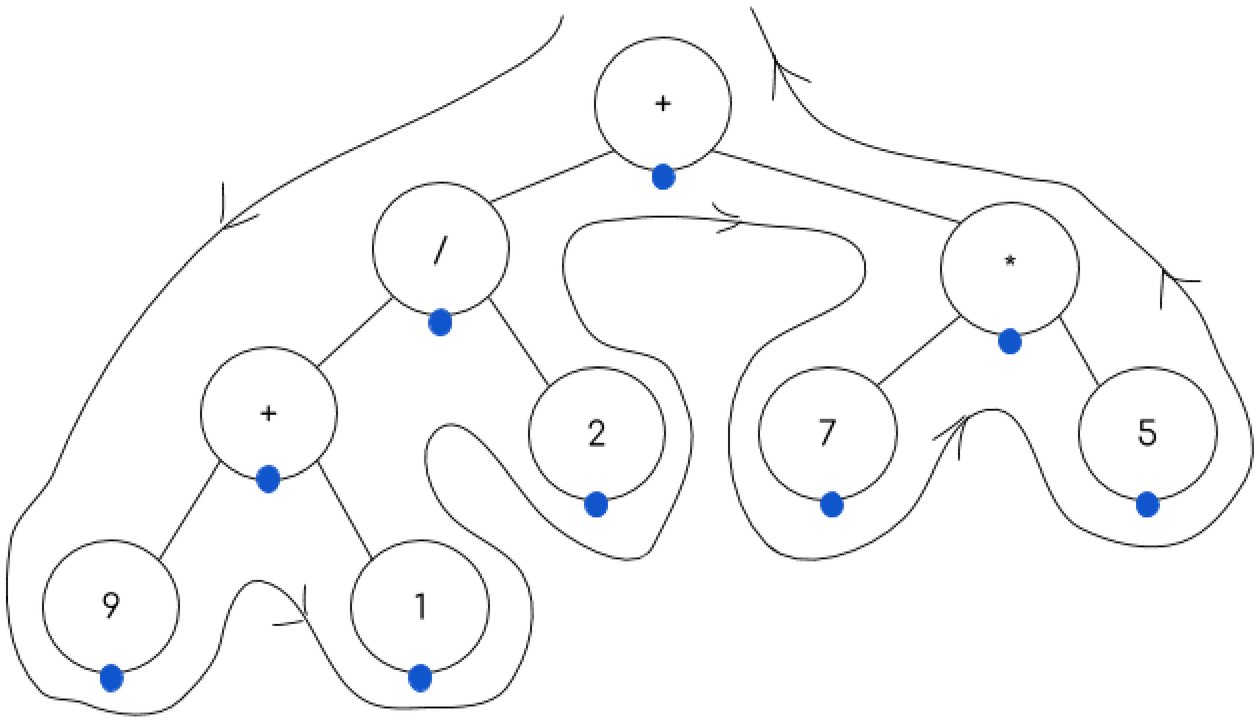
The first and second subtrees are completely atomised - they cannot be broken up any further. The third subtree has an operator in one of its child nodes. We can replace the 9 + 1 with the subtree for this equation.



In the fourth subtree, both child nodes had opcode in them. These nodes need to be replaced with the subtrees which represent their operation.



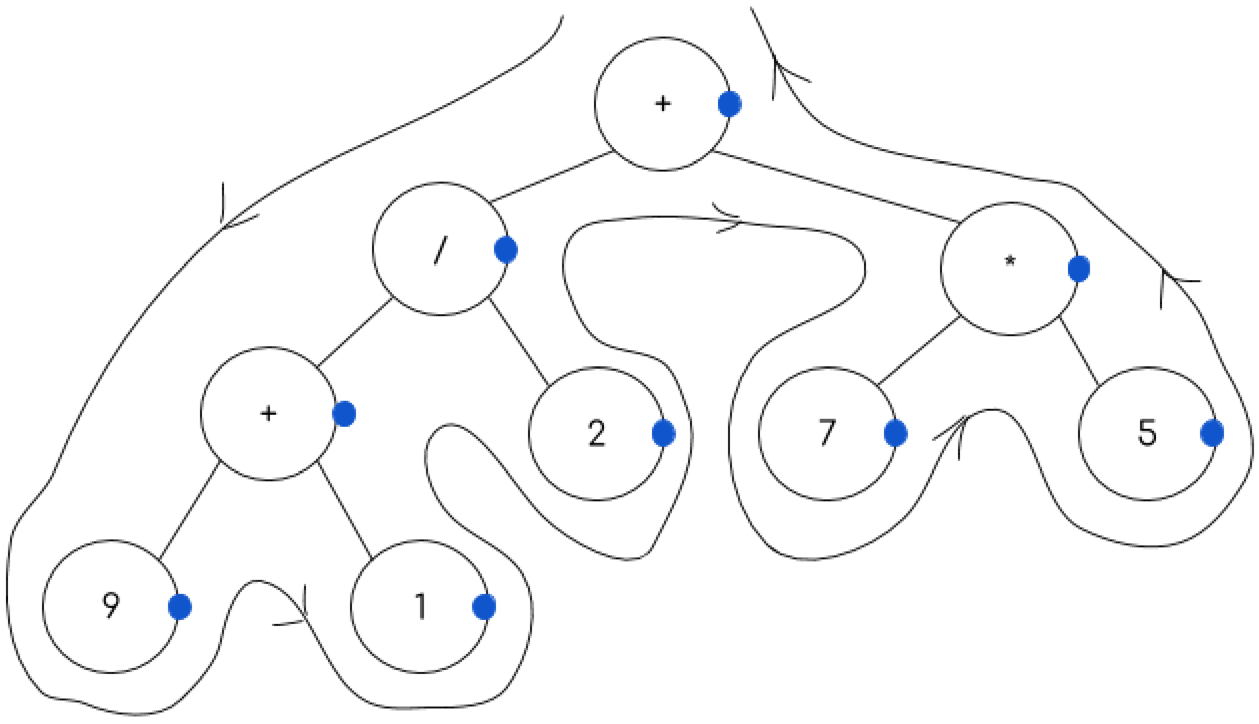
This is the expression tree. By doing an inorder traversal, the infix expression can be derived.



Output: 9 + 1 / 2 + 7 * 5

As you can see, the output produced by an inorder traversal leads to confusion over the order of execution. This problem could be mitigated by putting brackets around the operation of each node: (((9 + 1) / 2) + (7 * 5)).

By performing a postorder traversal, the postfix expression will be outputted.



Output: 9 1 + 2 / 7 5 * +

The postfix expression 9 1 + 2 / 7 5 * + is calculated as thus:

- When both of the operands are immediately preceding the opcode, the operation is performed. In the above example, this occurs twice, once at 9 1 + and at 7 5 *.
- 9 1 + is the same as 9 + 1 which equals 10
- 7 5 * is the same as 7 * 5 which equals 35
- Now the expression reads 10 2 / 35 +
- 10 and 2 are immediately before the operator /, so the equation 10 2 / is performed. It's infix equivalent is 10 / 2 which equals 5
- The expression reads 5 35 + ; there are two pieces of operand before the opcode so the equation 5 + 35 is solved which equals 40
- The answer is 40