Algorithm

Review: Data Structures

Instructor: Jae-Pil Heo (허재필)

Arrays

- An *array* is a data structure consisting of a collection of *values*, each identified by array *index*.

- In C/C++ programming language,
  - A consecutive set of memory location
  - Logical order is the same as the physical order
  - Creation of an array

```
Type d[10];
Type *d = new Type [ size ];
```

  - Accessing an element by array index

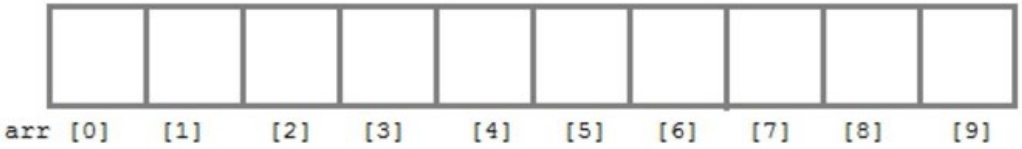
```
d[5] = 2;
```

  - Release the allocated memory

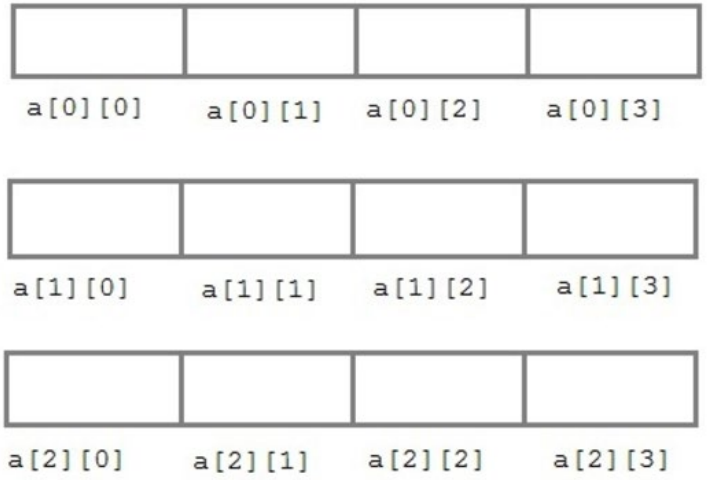
```
delete [] d;
```

Arrays

- 1 dimensional array
  - `int arr [10];`
  - `int *arr = new int [ 10 ];`



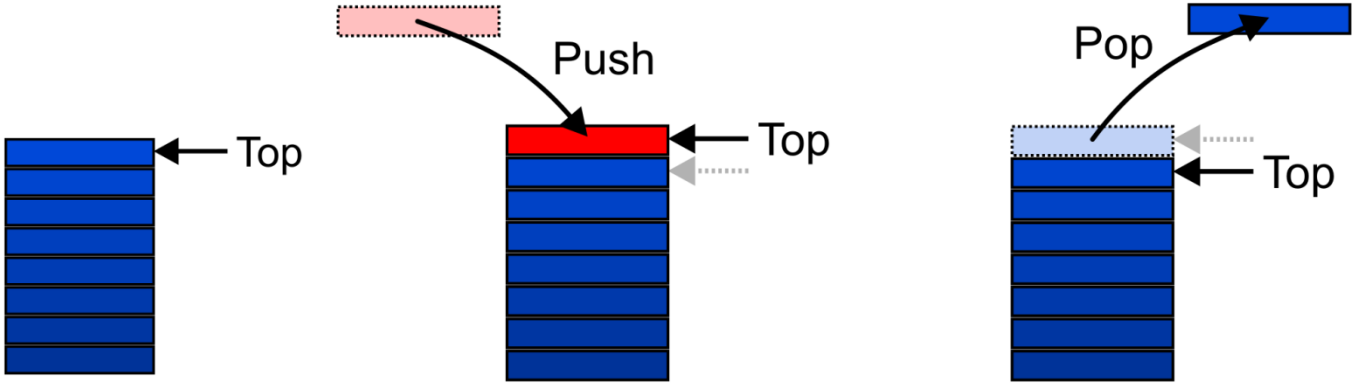
- 2 dimensional array
  - `int a[3][4];`
  - `int **a = new int * [3];`
    `for(int i=0;i<3;i++)`
    `{ a[i] = new int [4]; }`



Figures from http://www.studytonight.com/c/arrays-in-c.php

Stacks

- A stack has a *last-in-first-out (LIFO)* behavior
  - Insertion (push) and deletion (pop) are made at one position called ‘top’.

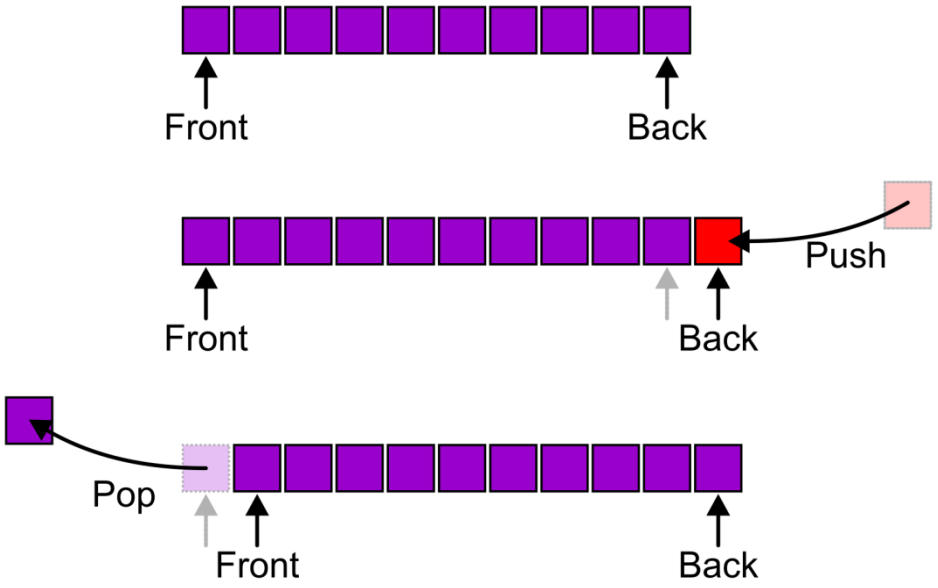


Applications of Stacks

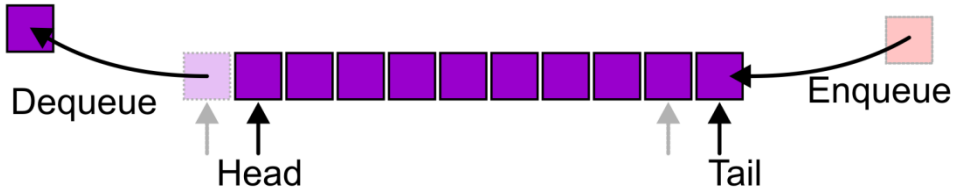
- Code parsing
  - Matching parenthesis
  - XML, HTML, …
- Tracking functions calls
- Dealing with undo/redo operations
- Tree/Graph traversal

Queues

- A queue has a first-in-first-out (FIFO) behavior.
  - Push (enqueue) is made at the position called ‘back (tail)’
  - Pop (dequeue) is made at the position called ‘front (head)’

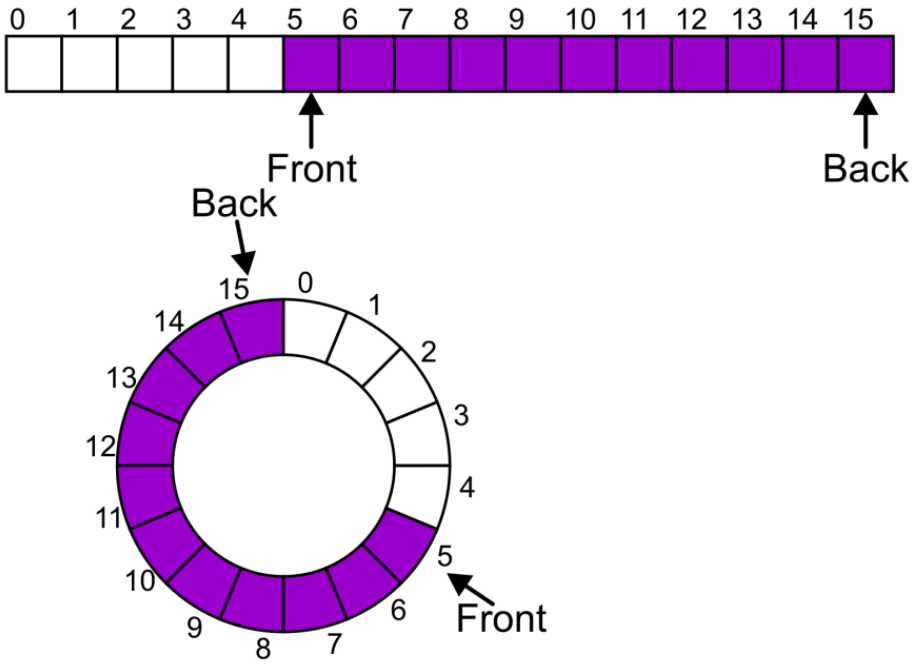


Alternative terms



Circular Queue

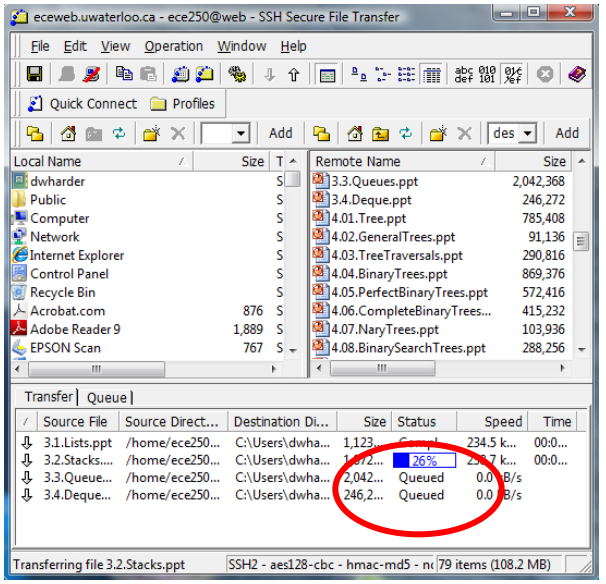
- Consider the indices being cyclic:



Applications of Queues

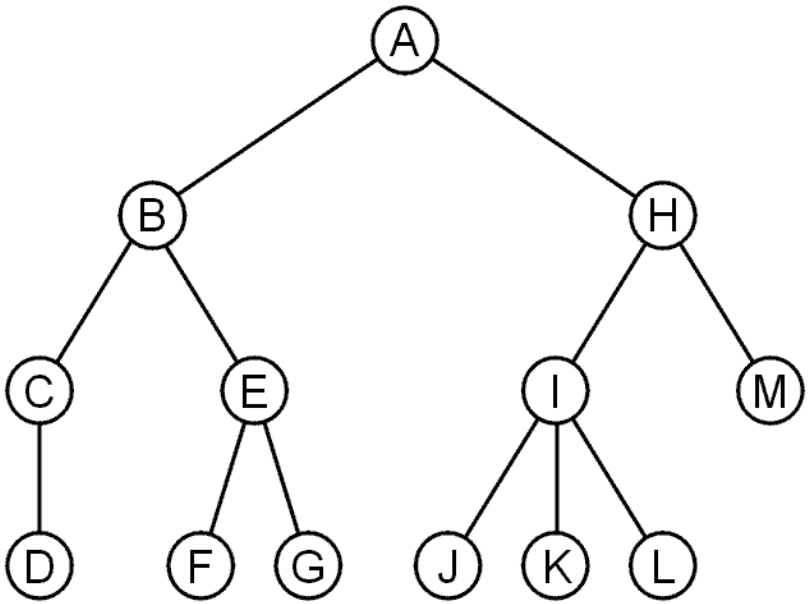
- Client-Server (client-casher) models
  - Multiple clients requesting services
  - Web, file ftp, database, mail, printer servers

- Breadth First Search (BFS)



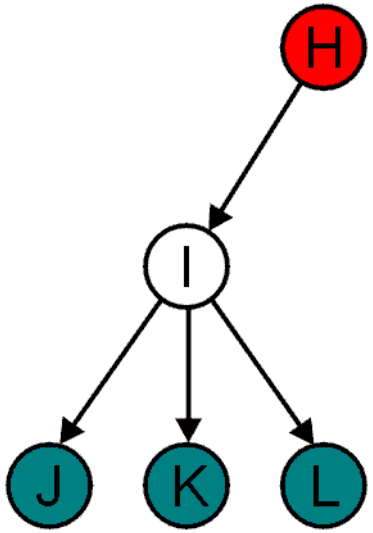
Trees

- A tree data structure stores information in nodes
  - There is a first node, or root
  - Each node has variable number of successors
  - Each node, other than the root, has exactly one node pointing to it



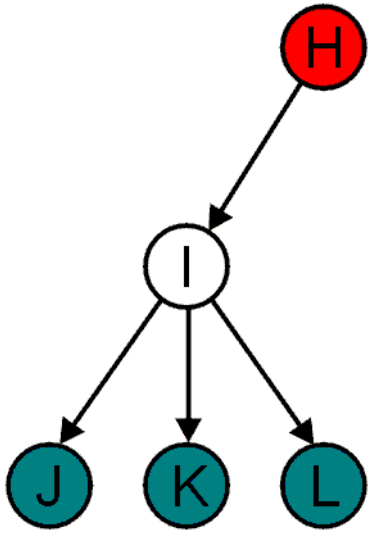
Trees

- All nodes will have zero or more child nodes or children
  - I has three children: J, K, and L
- For all nodes other than the root node, there is one parent node
  - H is the parent of I



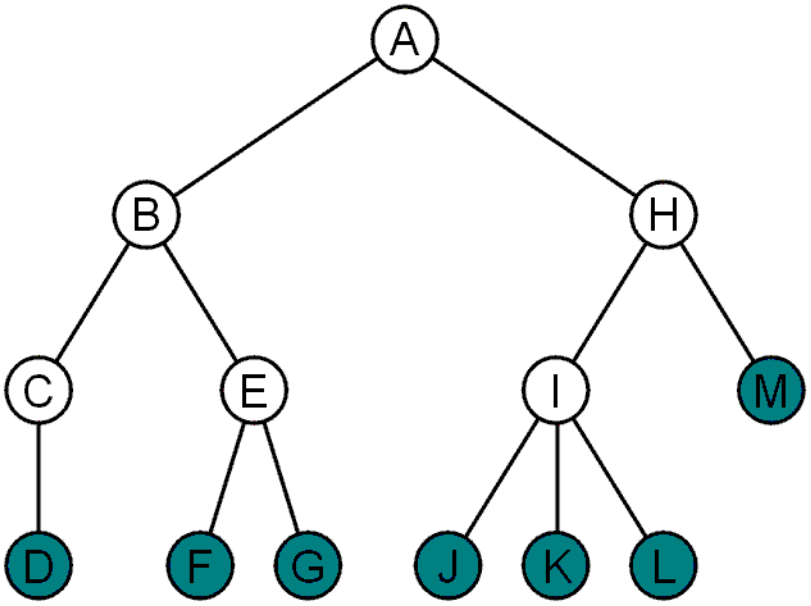
Trees

- The *degree* of a node is defined as the number of its children:
  - $\deg(I) = 3$
- Nodes with the same parent are *siblings*
  - J, K, and L are siblings



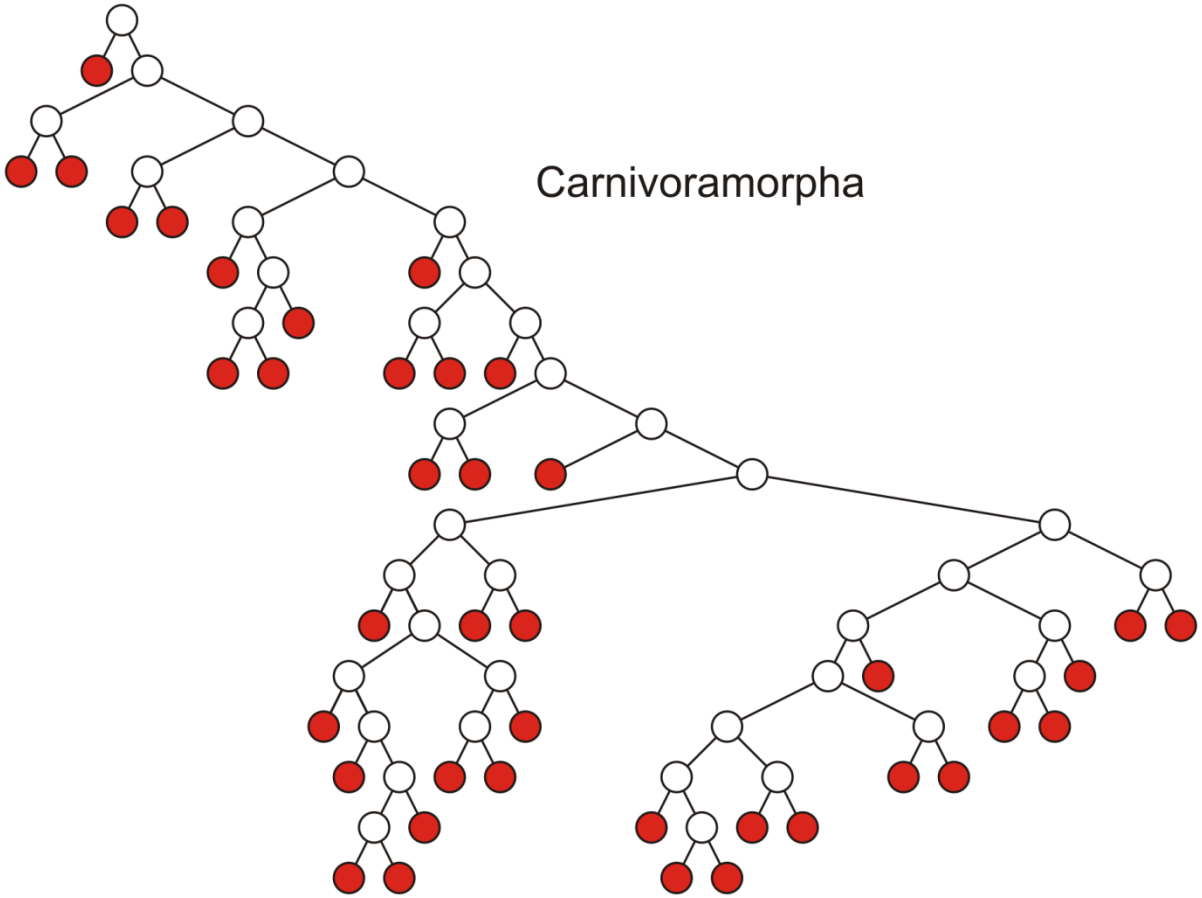
Trees (Leaf and Internal Nodes)

- Nodes with degree zero are also called *leaf nodes.*
- All other nodes are said to be *internal nodes*, that is, they are internal to the tree.



Trees (Leaf and Internal Nodes)

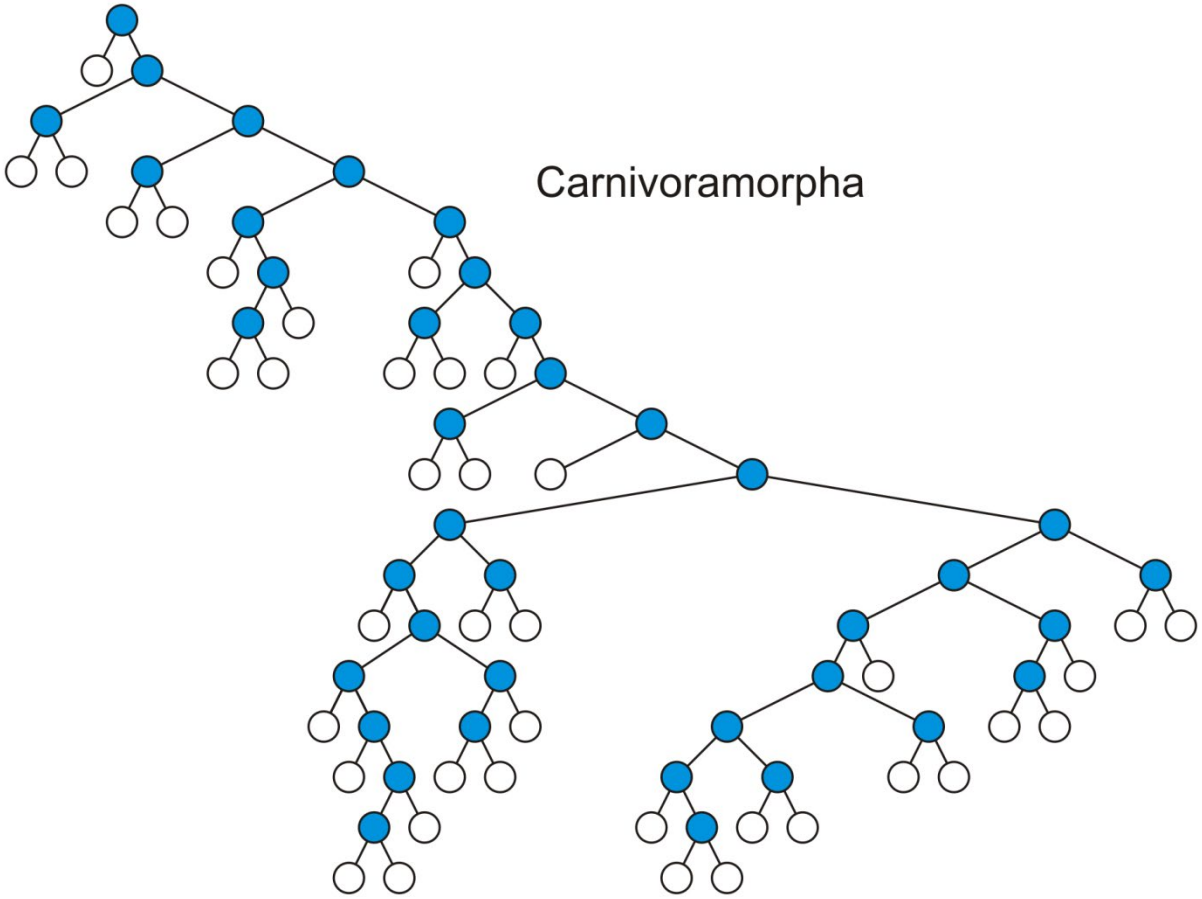
- Leaf nodes



Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

Trees (Leaf and Internal Nodes)

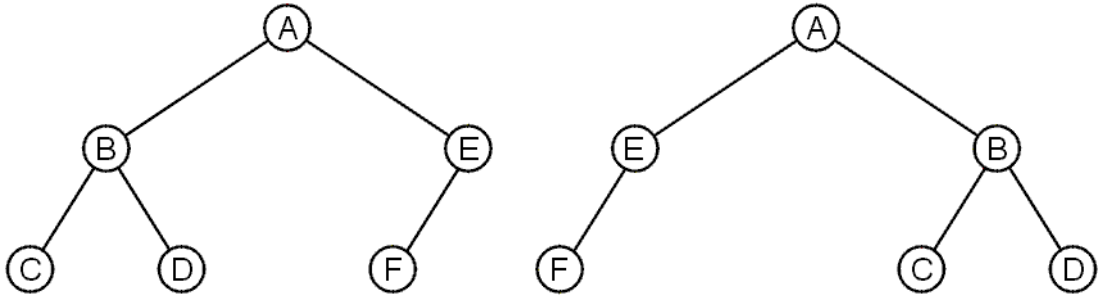
- Internal nodes



Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

Trees (Unordered & Ordered)

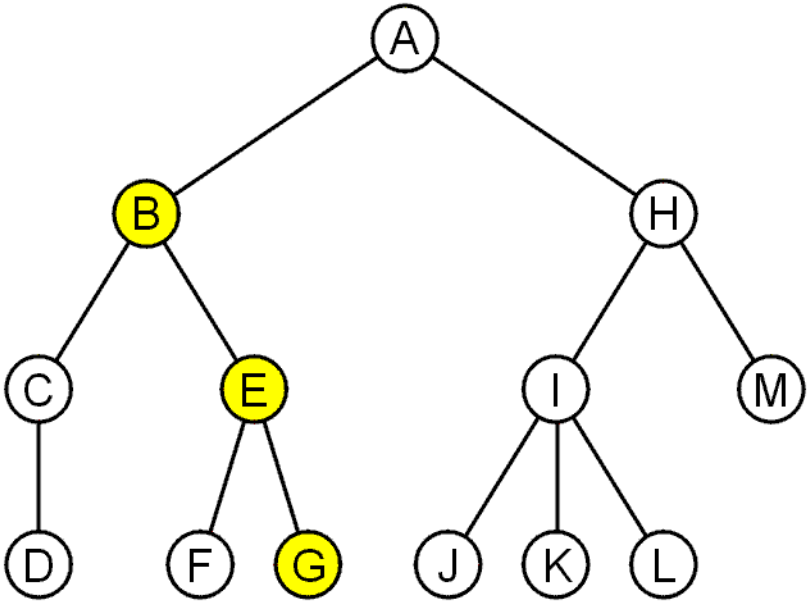
- These trees are equal if the order of the children is ignored
  - unordered tree



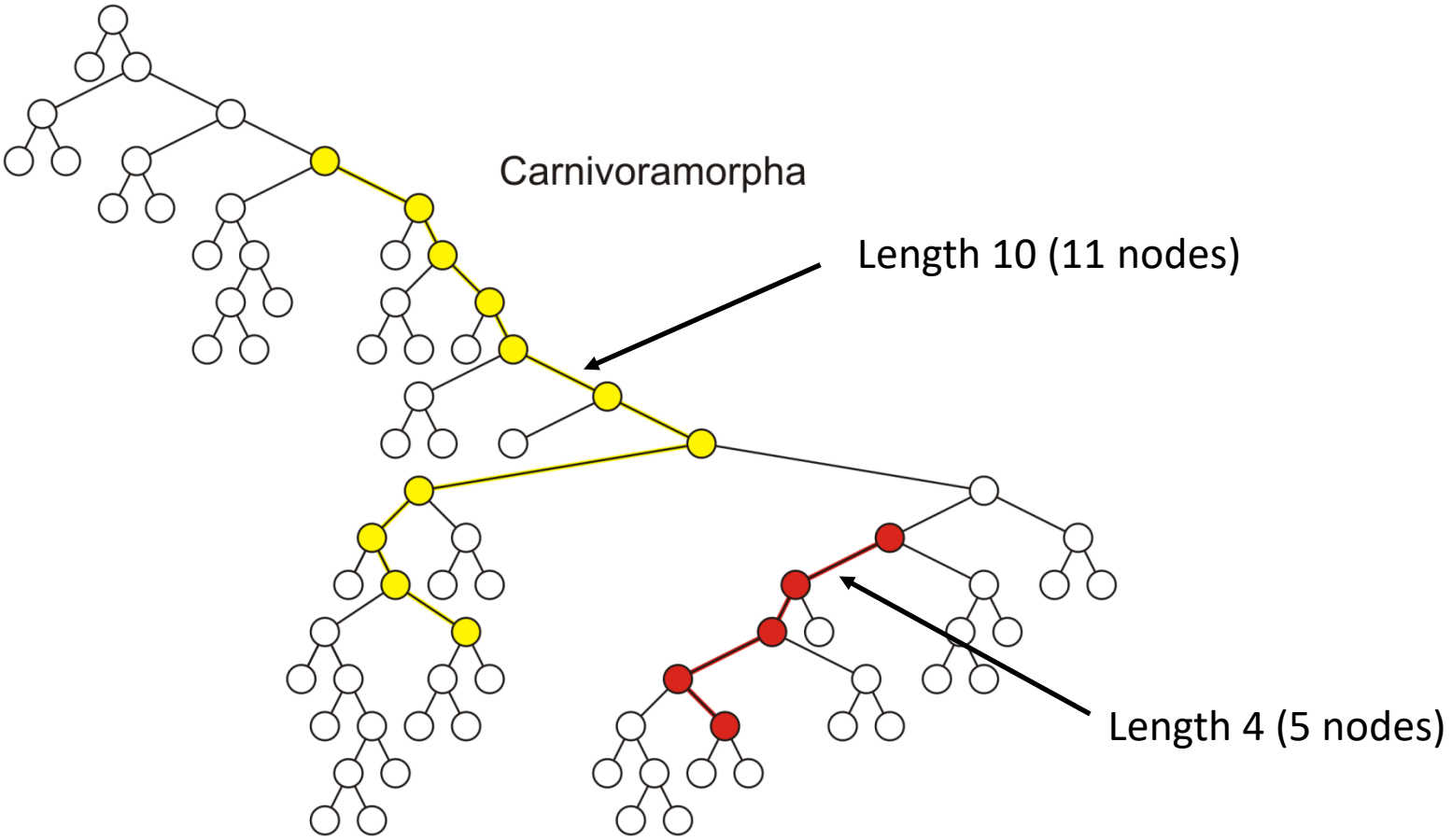
- They are different if order is relevant (ordered trees)

Trees - Paths

- A path is a sequence of nodes $(a_0, a_1, \ldots, a_n)$, where a_{k+1} is a child of a_k.
  - The length of this path is n
  - The path (B, E, G) has length 2



Trees - Paths

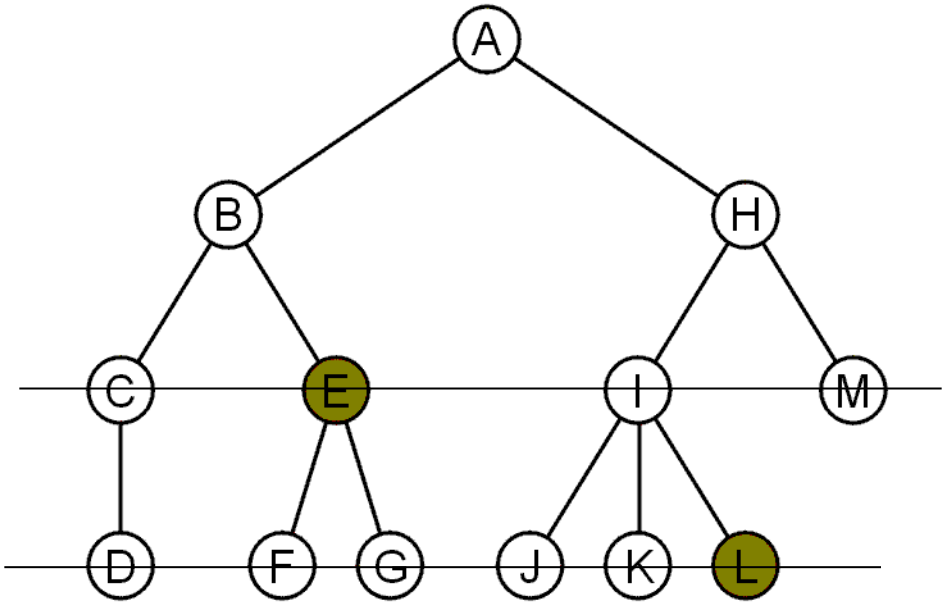


Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

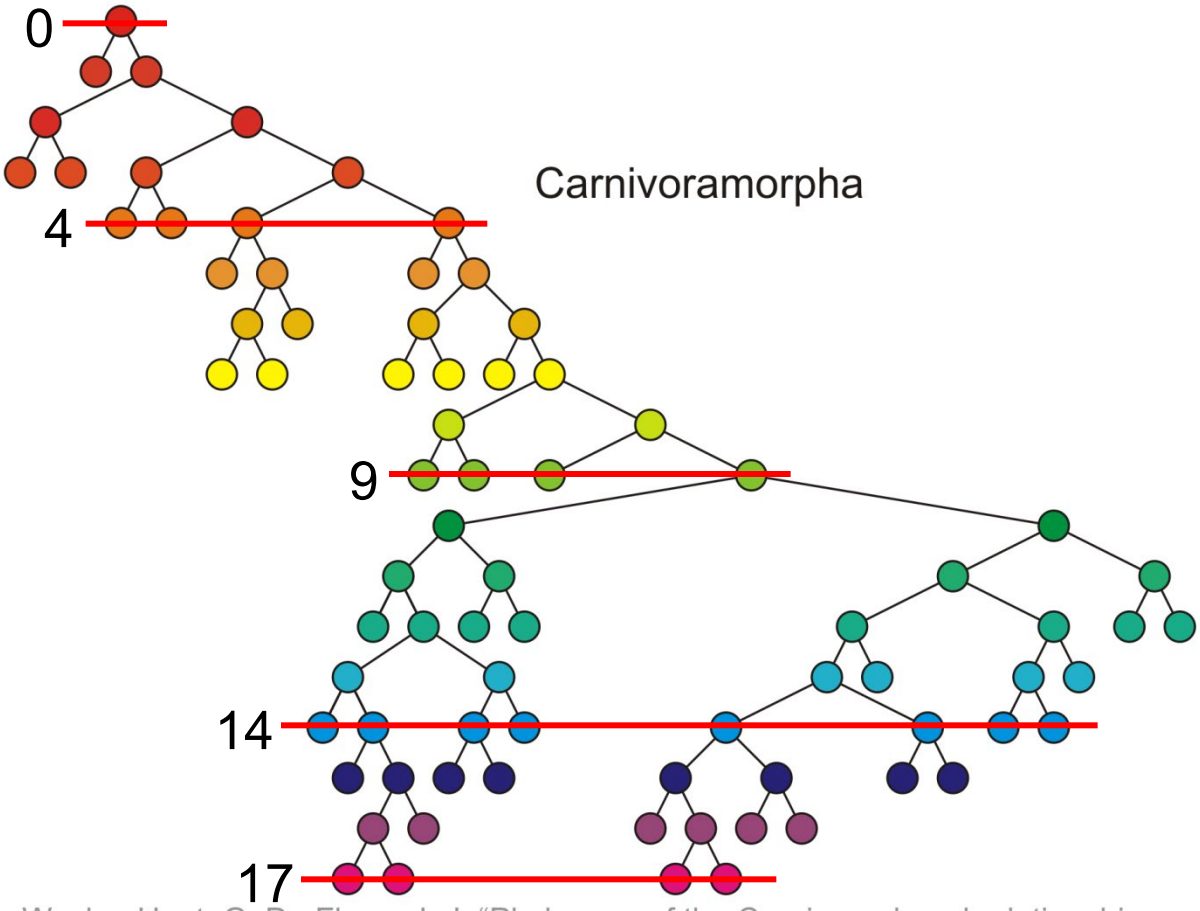
Trees - Depth

- For each node in a tree, there exists a unique path from the root node to that node.

- The length of this path is *depth* of the node
  - E has depth 2
  - L has depth 3



Trees - Depth



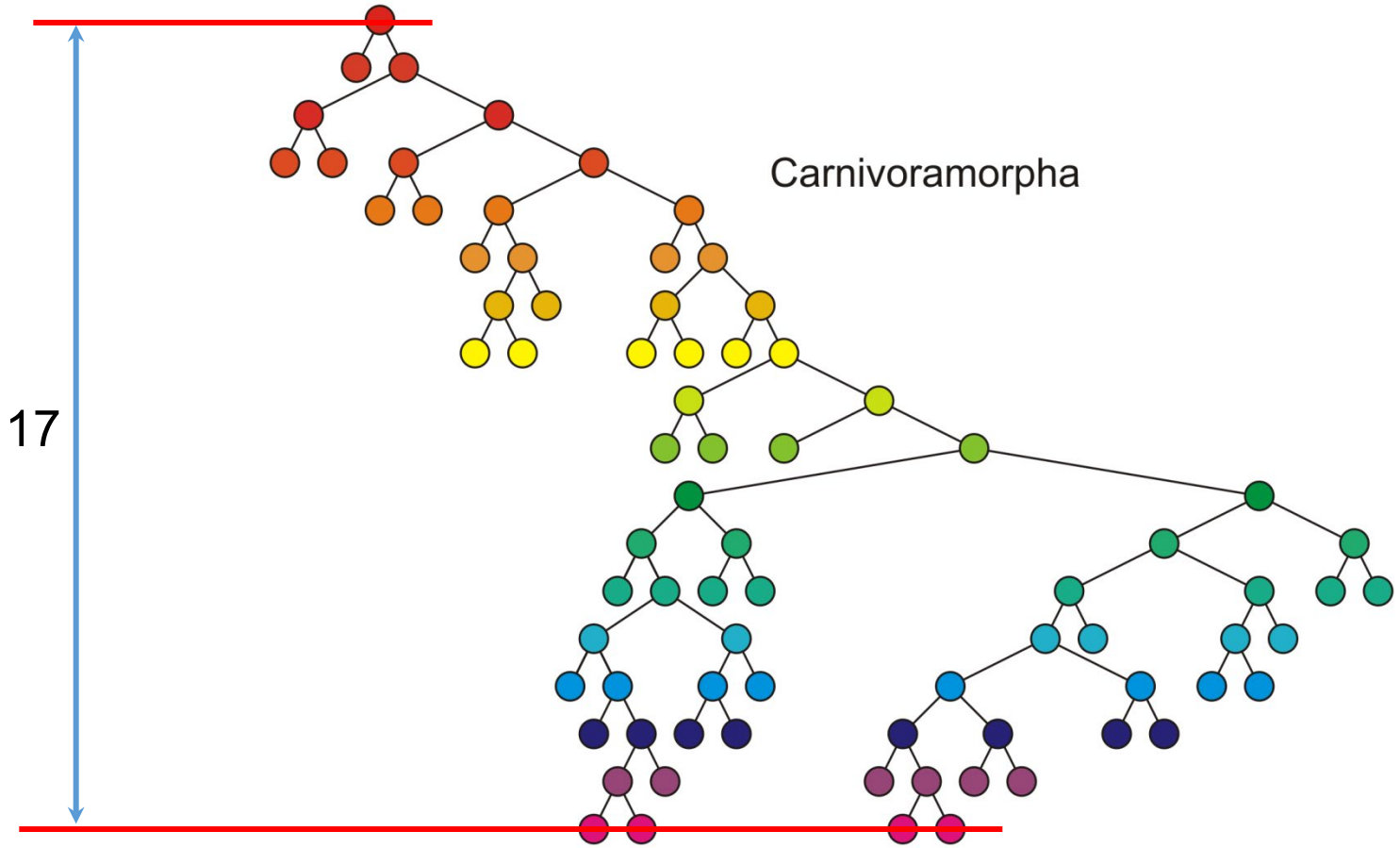
Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

Trees - Height

- The height of a tree is defined as the maximum depth of any node within the tree
  - The height of a tree with one node is 0
    - Just the root node
  - For convenience, we define the height of the empty tree to be -1

Trees - Height

- The height of this tree is 17



Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

Trees – Ancestor and Descendant

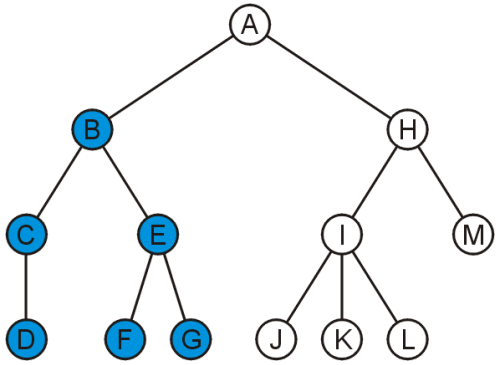
- If a path exists from node a to node b
  - a is an ancestor of b
  - b is a descendent of a

- Thus, a node is both an ancestor and a descendant of itself
  - We can add the adjective *strict* to exclude equality: a is a *strict descendent* of b if a is a descendant of b but $a \neq b$

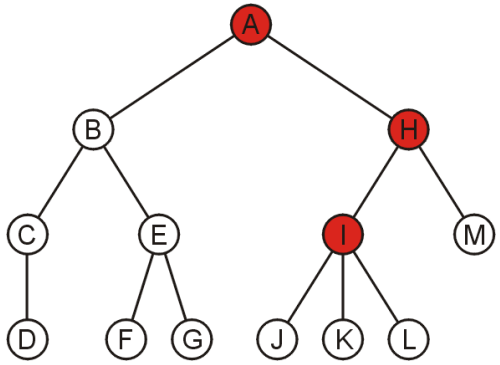
- The root node is an ancestor of all nodes

Trees – Ancestor and Descendant

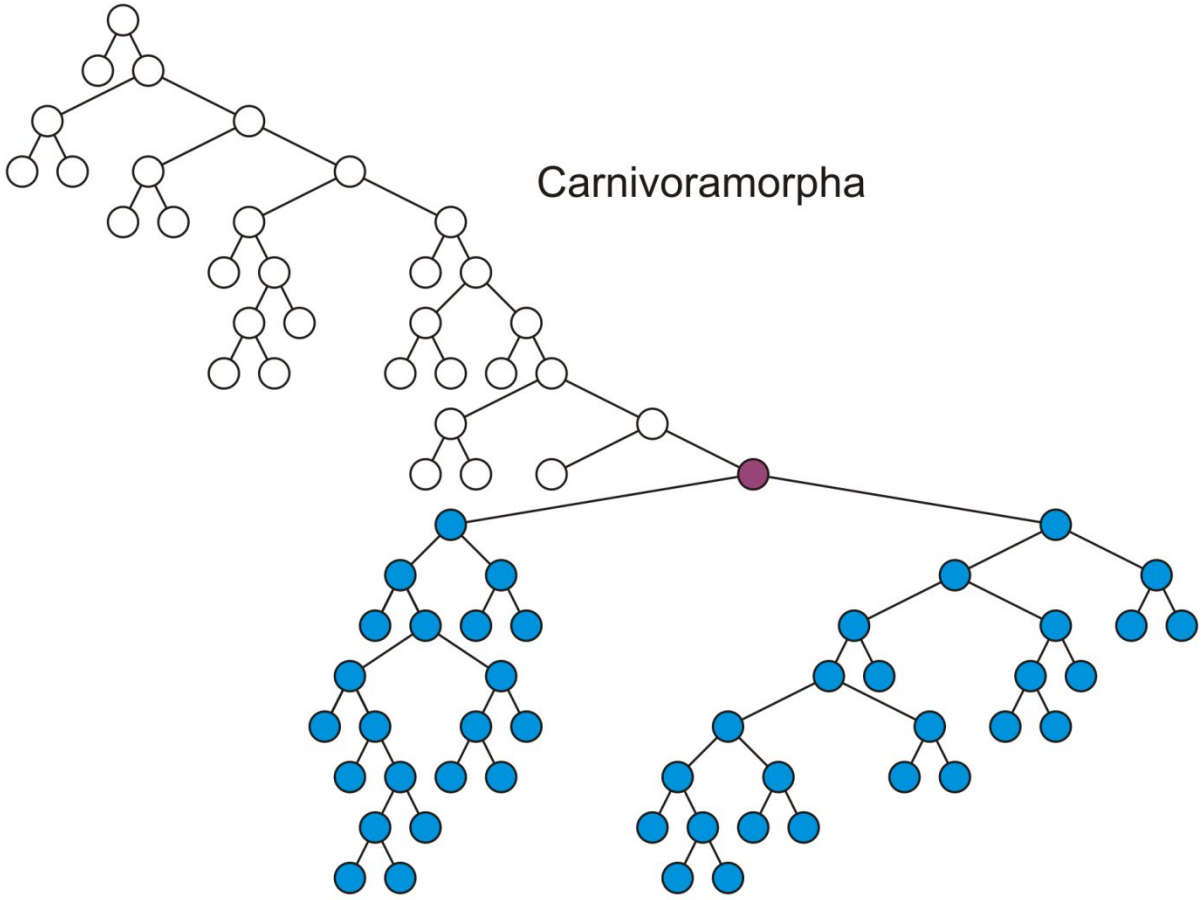
- The descendants of node B are B, C, D, E, F, and G:



- The ancestors of node I are I, H, and A:

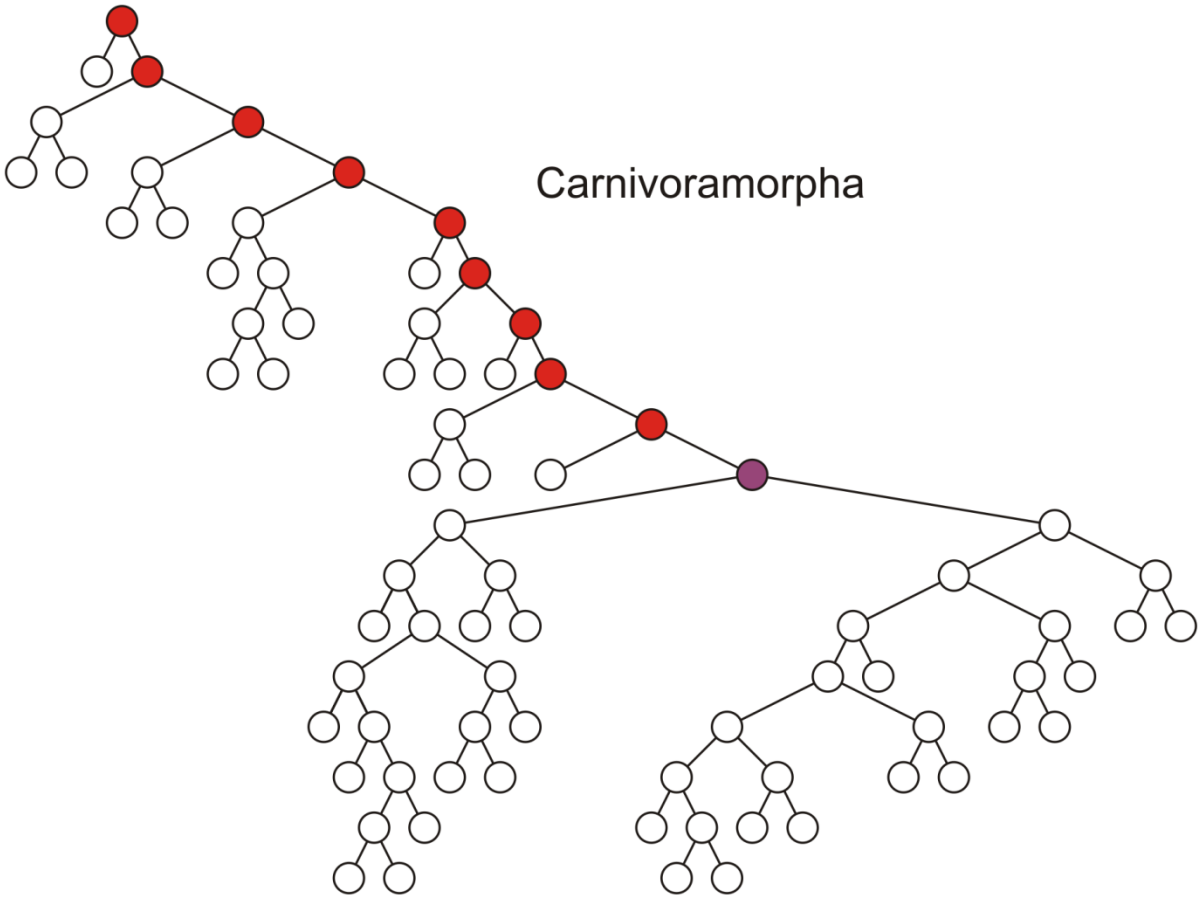


Trees - Descendants



Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

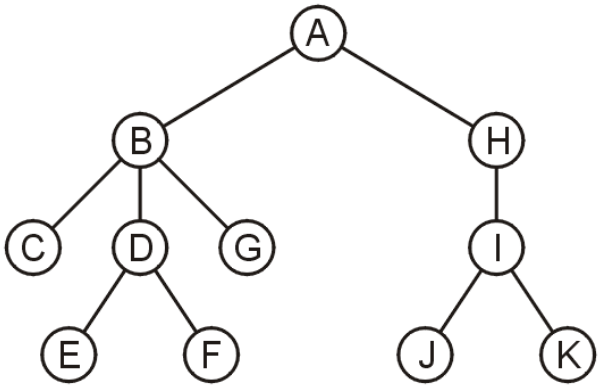
Trees - Ancestors

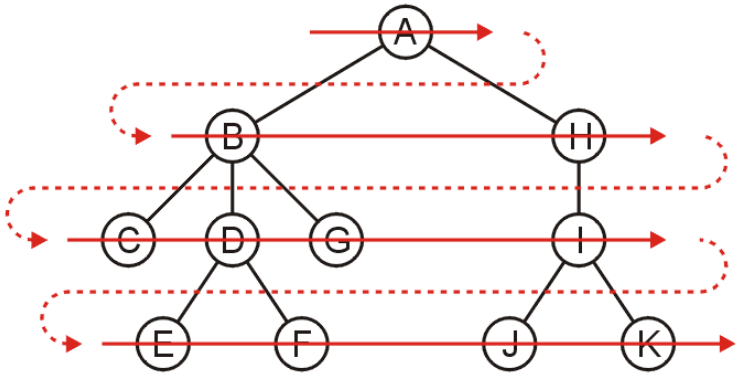


Wesley-Hunt, G. D.; Flynn, J. J. “Phylogeny of the Carnivora: basal relationships among the Carnivoramorphans, and assessment of the position of ‘Miacoidea’

Breadth First Search (BFS)

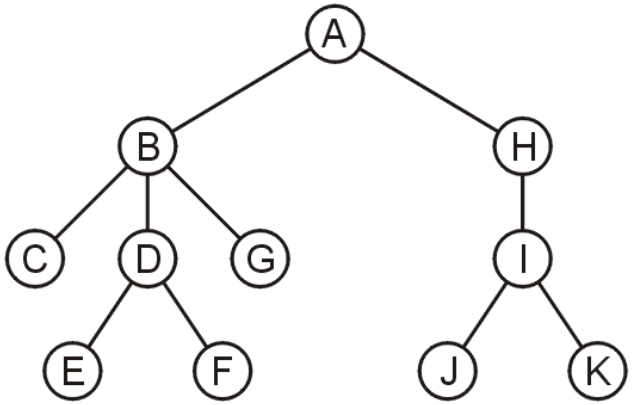
- Search all sibling nodes at one level before descending a level





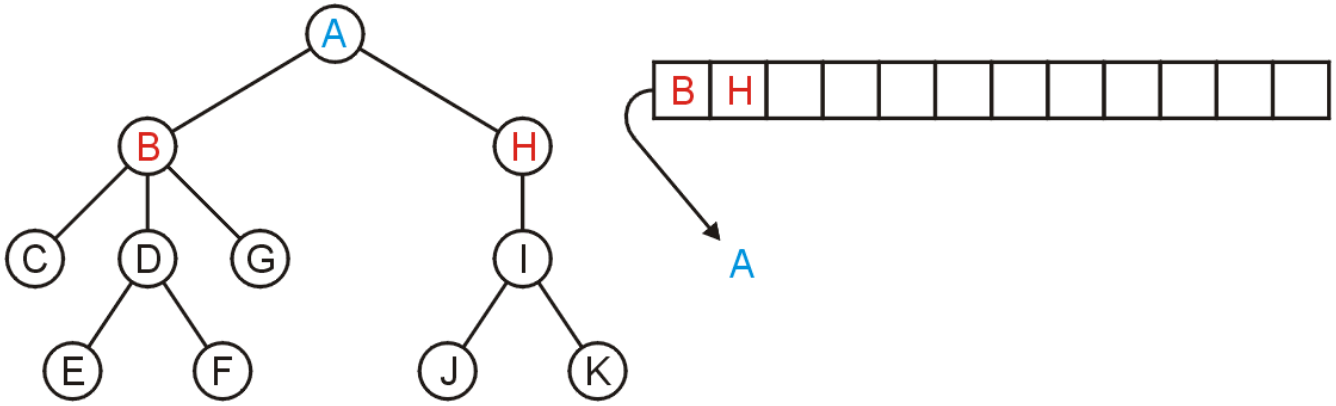
Breadth First Search using Queue

Push the root directory A



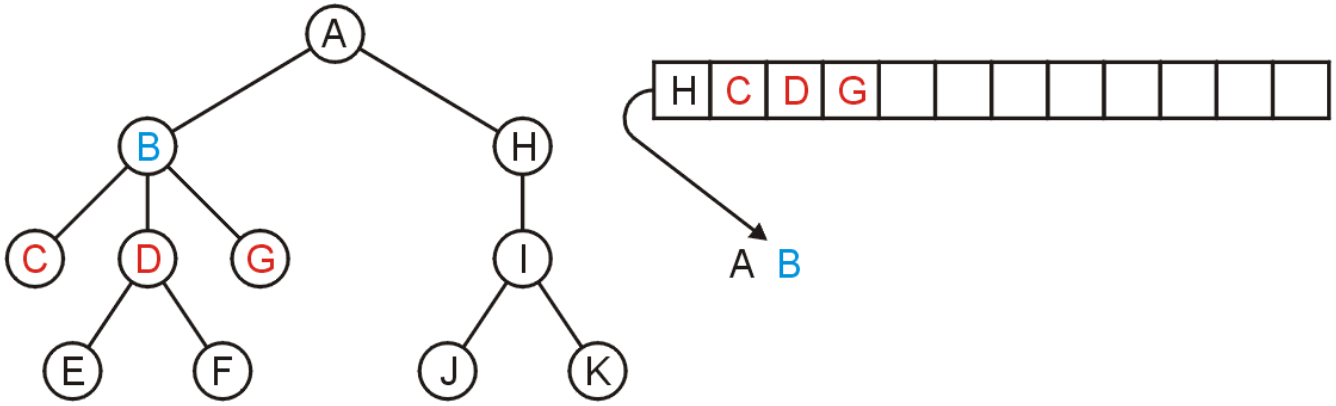
Breadth First Search using Queue

Pop A and push its two children: B and H



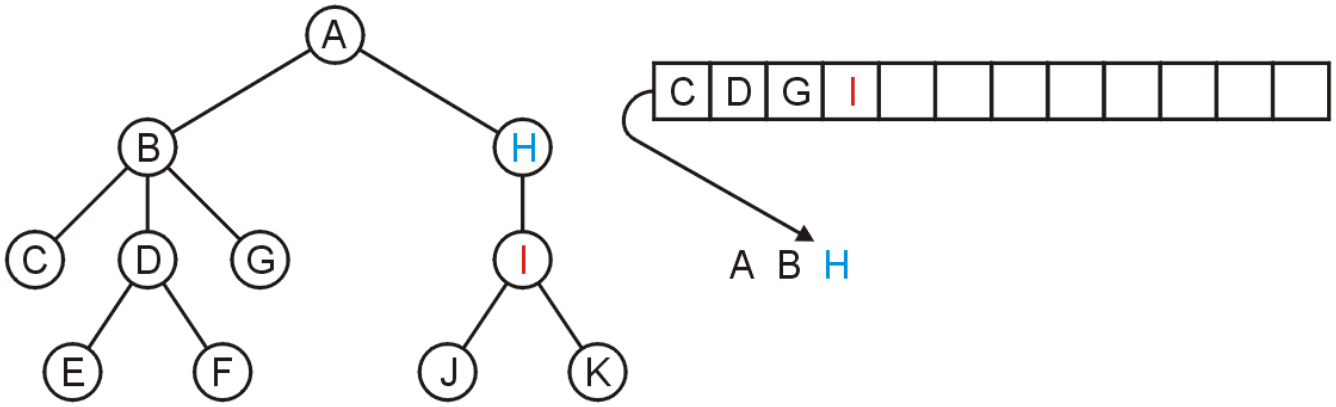
Breadth First Search using Queue

Pop B and push C, D, and G



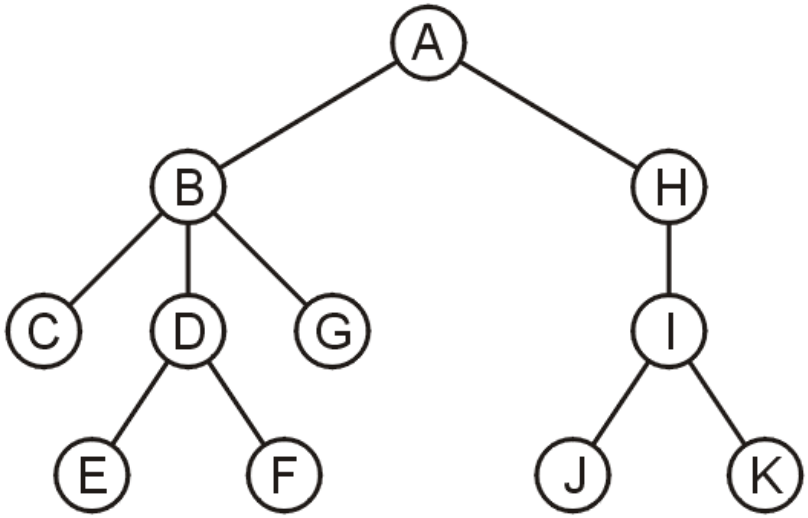
Breadth First Search using Queue

Pop H and push its one children I



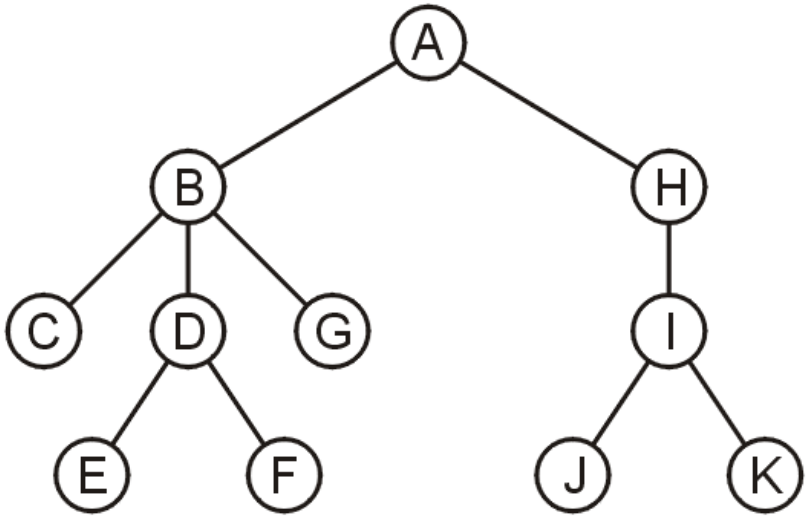
Breadth First Search using Queue

The resulting order: A B H C D G I E F J K is in breadth-first order



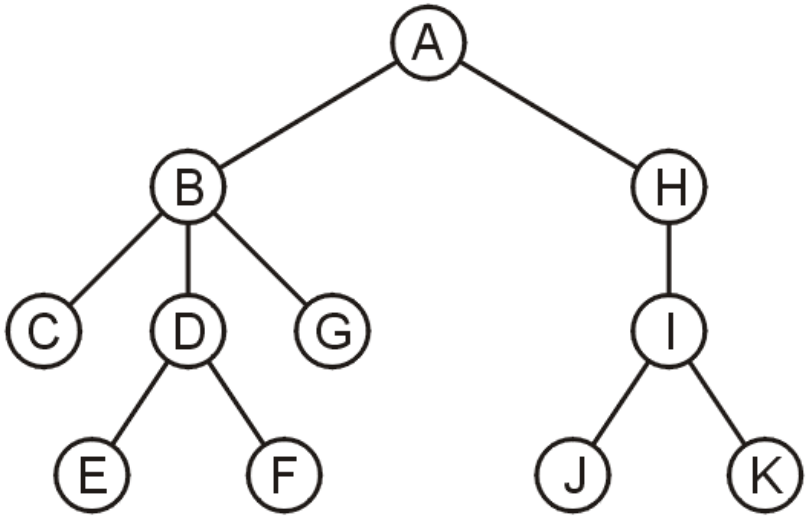
Breadth First Search using Queue

The resulting order: A B H C D G I E F J K is in breadth-first order



Depth First Search using Stack

The resulting order: A B C D E F G H I J K is in depth-first order



Any Question?