CSC 212: Data Structures and Abstractions
Spring 2018
University of Rhode Island

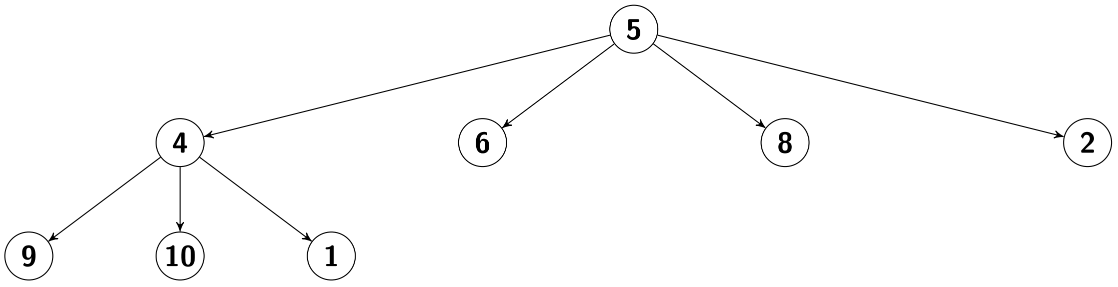
Weekly Problem Set #08 Solutions

Due Thursday 4/12 before class. Please turn in neat, and organized, answers hand-written on standard-sized paper **without any fringe**. At the top of each sheet you hand in, please write your name, and ID. The only library you're allowed to use in your answers is `iostream`.

1 K-ary Trees

1. Draw a k-ary tree, where `k=4`, after the insertion of the following elements in order: *Assuming insertions are performed left to right, level by level*

   `[5, 4, 6, 8, 2, 9, 10, 1]`



2. Looking at the tree you have drawn, how many leaves and nodes are present?

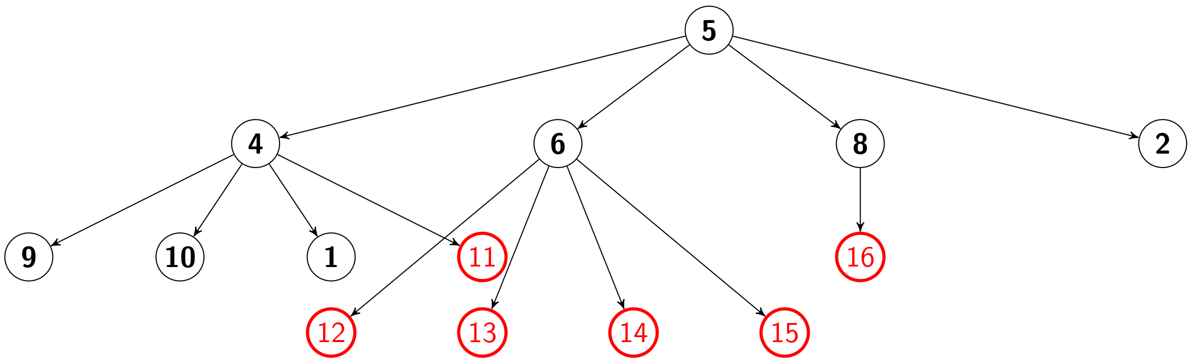
   6 leaves, 8 nodes, 2 of which are internal.

3. Examine your tree and find both the root and the node with the value 4. For both nodes, list the following attributes: depth, height of subtrees, number of siblings, number of children.

   Root Node: { depth: 0, height of subtrees: [2, 1, 1, 1], number of siblings: 0, number of direct children: 4, descendants: 7 }

   Node(4) { depth: 1, height of subtrees: [1, 1, 1, 0], number of siblings: 3, number of direct children: 3, descendants: 3 }

4. Insert 6 more random elements into your tree and relist any of the above attributes that have changed.



   Root Node: { depth: 0, height of subtrees: [2, 2, 2, 1], number of siblings: 0, number of direct children: 4, descendants: 13 }

Node(4) { depth: 1, height of subtrees: [1, 1, 1, 1], number of siblings: 3, number of direct children: 4, descendants: 4 }

5. Would the structure (shape of the tree and not values of the nodes) of the k-ary tree you've drawn change at all if the elements were inserted in sorted order? Explain why or why not.

   No. K-ary trees are not sorted trees.

2 Doubly Linked Lists

1. Describe an algorithm to count the number of sets of three adjacent nodes whose sum is equal to 0.
   Set a counter to 0. Set a pointer to the second element of the list. Check whether the sum of the node pointed to, its predecessor, and its successor is 0 and increment the counter if true. Set the pointer to the next element of the list and repeat until the successor is null.
2. For a doubly linked list with n elements, how much additional memory is going to be used compared to a singly linked list?
   n pointers will be allocated

3 Stacks and Queues

1. Is a linked list the best underlying structure to implement a queue with? Justify your answer.

   Linked lists are a convenient underlying structure, but there is no true best when it comes to queues. So long as your implementation offers constant time operations and minimal space requirements.
2. Is a linked list the best underlying structure to implement a stack with? Justify your answer.

   Linked lists are a convenient underlying structure, but there is no true best when it comes to queues. So long as your implementation offers constant time operations and minimal space requirements.
3. Would a stack or queue be more efficient for the following:
   (a) An undo button in a text editor
   stack
   (b) A web server
   queue
   (c) A breadth-first search
   queue
   (d) A depth-first search
   stack