**School of Computer Science**

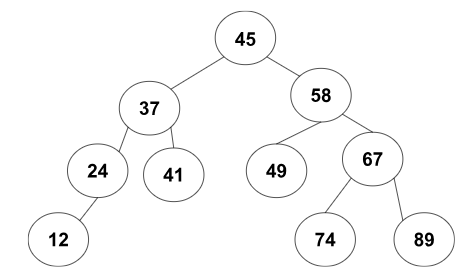
**CIS\*2520: Data Structures**

**Fall 2024, Lab 6**

**Week of Nov. 11 to Nov. 15**

# 1 Trees

1. Given the binary tree below



1. What is the root node?

Answer: 45

1. List all Leaf Nodes

Answer: 12, 74, 89

1. What is the height of the binary tree?

Answer: 4

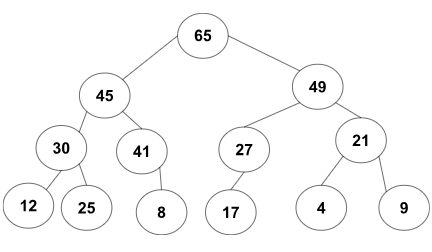
1. What is the degree of the node with a value of 74?

Answer: 0

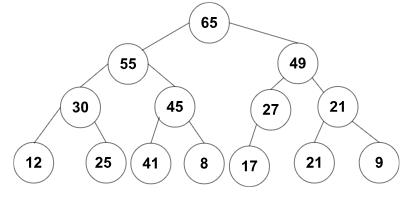
1. Traversal
2. Pre Order Traversal of the tree Answer: 45, 37, 24, 12, 41, 58, 49, 67, 74, 89
3. In Order Traversal of the tree Answer: 12, 24, 37, 41, 45, 49, 58, 74, 67, 89
4. Post Order Traversal of the tree Answer: 12, 24, 41, 37, 49, 74, 89, 67, 58, 45
5. Level Order Traversal of tree Answer: 45, 37, 58, 24, 41, 49, 67, 12, 74, 89
6. Which traversal gives a sorted version of a tree? Answer: Inorder

# 2 Heaps

1. Given the following heap

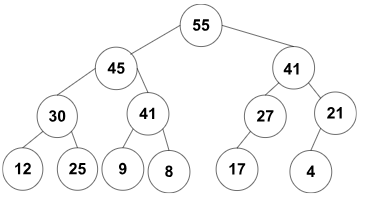


1. determine if it is a max heap or min heap. Explain your answer.  
   Answer: Max Heap because every parent node is greater than or equal to its children
2. In this heap, insert elements 55, show where the insertion location is, and the steps to restore the heap



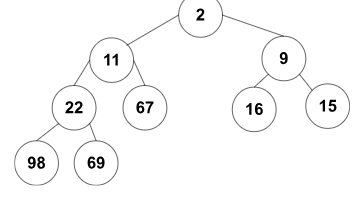
1. Insert 55 as end node, making it child of 41
2. 55 is greater than 41, which violates the max-heap property. So, swap 55 with 41
3. New parent is 45, but 55 is greater than 45, so we swap
4. New parent is 65, 55 is less than 65, so the max-heap property is satisfied

3) Following the previous step (after the insertion of 55), remove 65, and show the steps to restore the heap



1. Replace 65 with right most node 9, remove 65
2. As 9 is less than both childern, we swap with 55 the largest child. The new root node
3. We swap with 45 as its the largest child and then again with 41

4. Given an array of numbers, construct a min-heap. Show your steps. Array = [15,98,9,67,22,16,2,11,69].



1) Make 15 as root 2) Insert 98 as left child of 15 3) Now insert 9 as right child of 15, but 9 is smaller so swap them 4) Insert 67 as left child of 98, but 67 is smaller so swap them 5) Insert 22 as right child of 67, but 22 is smaller so we swap 6) Insert 16 as left child of 15, no swaps needed 7) Insert 2 as right child of 15, we swap as 2 is smaller, we swap again with 9, making 2 the root and 9 being the parent of 16 and 15 8) Insert 11 as left child of 98 swap 11 with 98 as its smaller and then again with 22 making it a parent of 22 and 67 9) Then you can insert 69 as right child of 22

5. Write pseudocode for heapsort

def heapify(arr, n, i):

largest = i # Initialize largest as root

left = 2 \* i + 1 # Left child index

right = 2 \* i + 2 # Right child index

# If left child exists and is greater than root

if left < n and arr[left] > arr[largest]:

largest = left

# If right child exists and is greater than current largest

if right < n and arr[right] > arr[largest]:

largest = right

# If largest is not the root, swap and continue heapifying

if largest != i:

arr[i], arr[largest] = arr[largest], arr[i] # Swap root and largest

heapify(arr, n, largest) # Recursively heapify the affected subtree

def heap\_sort(arr):

n = len(arr)

# Build a max heap

for i in range(n // 2 - 1, -1, -1):

heapify(arr, n, i)

# One by one, extract elements from the heap

for i in range(n - 1, 0, -1):

arr[0], arr[i] = arr[i], arr[0] # Swap current root with the end element

heapify(arr, i, 0) # Call max heapify on the reduced heap