**Dayu Liu**

**Sorting:**

A

1. What would be the complexity of Quicksort if the pivot is always the median? (Assume the median can always be found in linear time).
   1. Always O(n log n)
   2. Always O(n2)
   3. Worst case: O(n2), average case: O(n log n)
   4. Worst case: O(n2 log n), average case: O(n2)

Average is O(nlogn) because tree height is logn and we go through n elements. Because the pivot is always the median, there is no worst case, it’s always O(n log n).

D

1. Which of the following statements about sorting is incorrect?
   1. Insertion sort is more efficient on a partially sorted array.
   2. Mergesort has the best worst-case time complexity for comparison-based sorting algorithms.
   3. Radix sort’s running time depends on the keys and base.
   4. For large n, Heapsort is less efficient than insertion sort.

Insertion sort is O(n^2), not good for large n

C

1. Given a pre-sorted array of unique integers, what is the expected running time of Quicksort if the pivot is always chosen to be either the first or last element?
   1. O(n log n)
   2. O(n2 log n)
   3. O(n2)
   4. None of the above

The height will be n if we pick margin element every time and we have n elements, so O(n^2)

A

1. What would be the complexity of Mergesort if the list is divided into 4 chunks instead of 2 each time?
   1. O(n log n)
   2. O(n2)
   3. O(n3/4)
   4. O(n log2 n)

N\*Log4n = n \* (log4/log2) \* log2n = 2n\* log2n, which is still O(nlogn)

1. Show the sorting steps of Radix sort for the following input:  
    25, 57, 48, 37, 12, 92, 86, 33

Sort numbers into buckets using 1st digit from right

Bucket 0-

Bucket 1-

Bucket 2-12,92

Bucket 3-33

Bucket 4-

Bucket 5-25

Bucket 6-86

Bucket 7-57,37

Bucket 8-48

Bucket 9-

Take Numbers out from bucket 0-9 from bottom to top

12,92,33,25,86,57,37,48

Sort numbers into buckets using 2st digit from right

Bucket 0-

Bucket 1-12

Bucket 2-25

Bucket 3-33,37

Bucket 4-48

Bucket 5-57

Bucket 6-

Bucket 7-

Bucket 8-86

Bucket 9-92

Now we take them out, they are in ascending order

12,25,33,37,48,57,86,92

1. Trace the Mergesort algorithm on the following array of integers. Be sure to show all split and merge steps.  
    23, 39, 59, 26, 30, 62, 27, 83

Left->right

Recursive calls of partition

23, 39, 59, 26, 30, 62, 27, 83

23, 39, 59, 26 ||| 30, 62, 27, 83

23, 39 || 59, 26 ||| 30, 62 || 27, 83

23| 39 || 59| 26 ||| 30| 62 || 27| 83

Recursive calls of merge

23, 39 || 26, 59 ||| 30, 62 || 27, 83

23,26,39,59 ||| 27,30,62,83

23,26,27,30,39,59,62,83

**Trees:**

A

1. Which of the following statements about trees is most accurate?
   1. The purpose of AVL trees is to efficiently implement search algorithms.
   2. The order of search algorithms in all binary trees is the same
   3. The depth of a binary tree is as most O(log n)
   4. The balance factor of AVL trees is 0 or 1.

Balance factor = heightOfLeftSubtree – heightOfRightSubtree, Can be -1

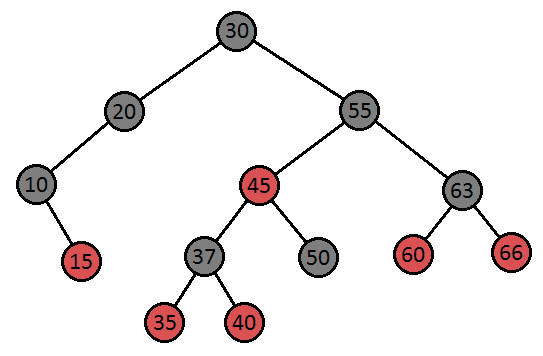
Depth is best case Ologn

Not the same

A

1. True or false: a Red-Black tree with 128 nodes has at least one red node.
   1. True
   2. False

If all nodes are black, all leaves won’t have the same black depth because there is a last leaf will be 1 depth larger.

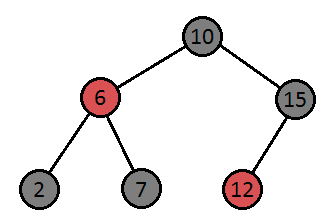
1. Is the following Red-Black tree (with leaf nodes hidden) valid? If not, which property is violated?

Yes, it is valid.

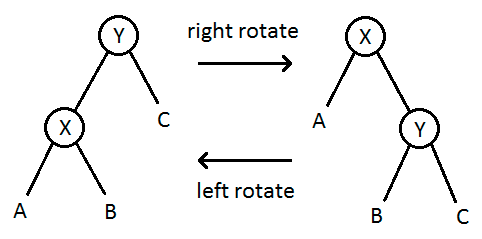
There are no double reds and root node is black and every path to one leaf has same numbers of black nodes, which is 3 here.

D

1. Consider the following Red-Black tree (with leaf nodes hidden):



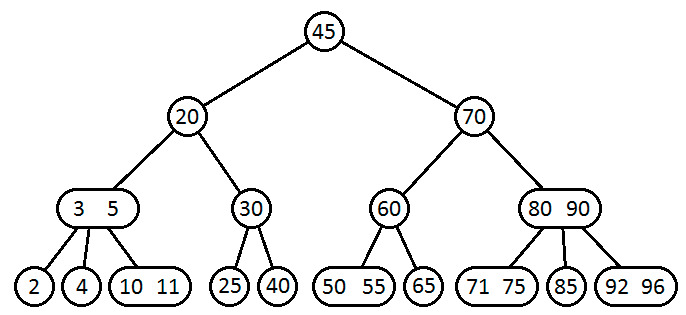
Suppose we want to insert the element 14. Which of the following is true?  
 For reference:

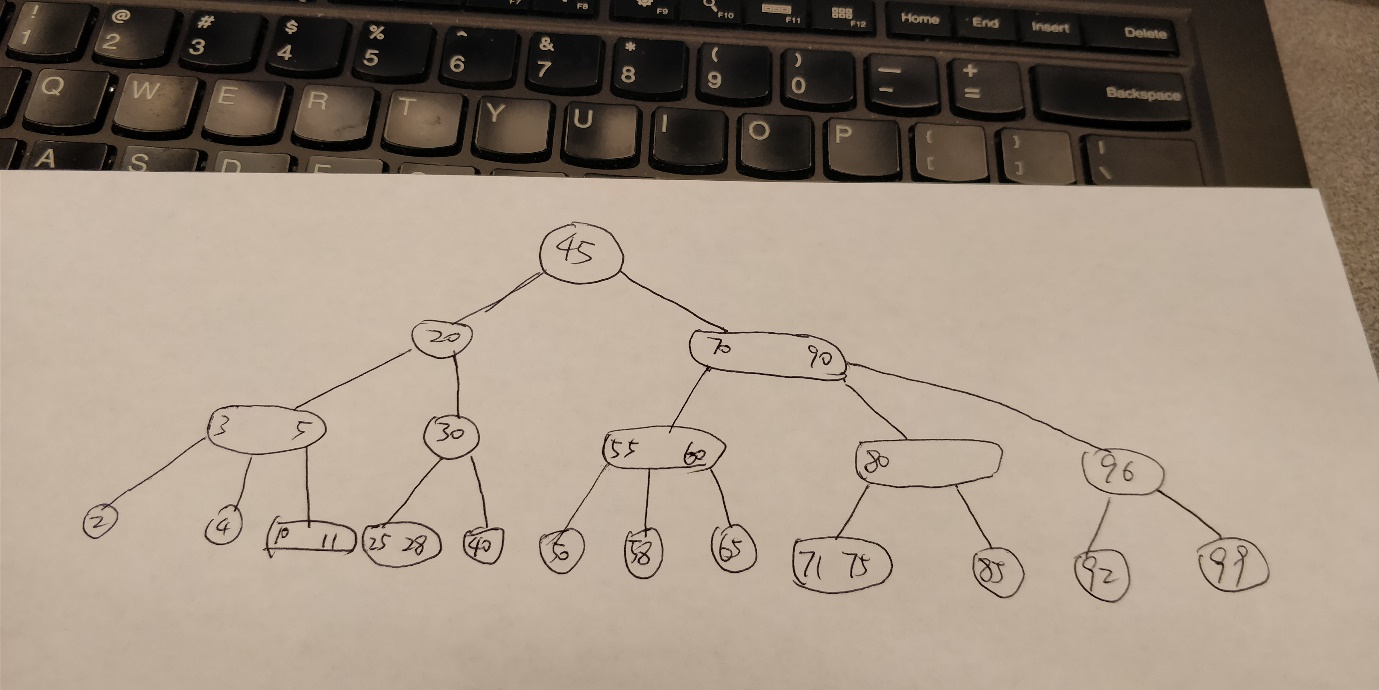


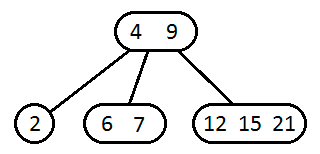
* 1. We must perform a recoloring only
  2. We must perform a left rotation only
  3. We must perform a right rotation only
  4. We must perform a left rotation followed by a right rotation
  5. The element can be inserted without any further action

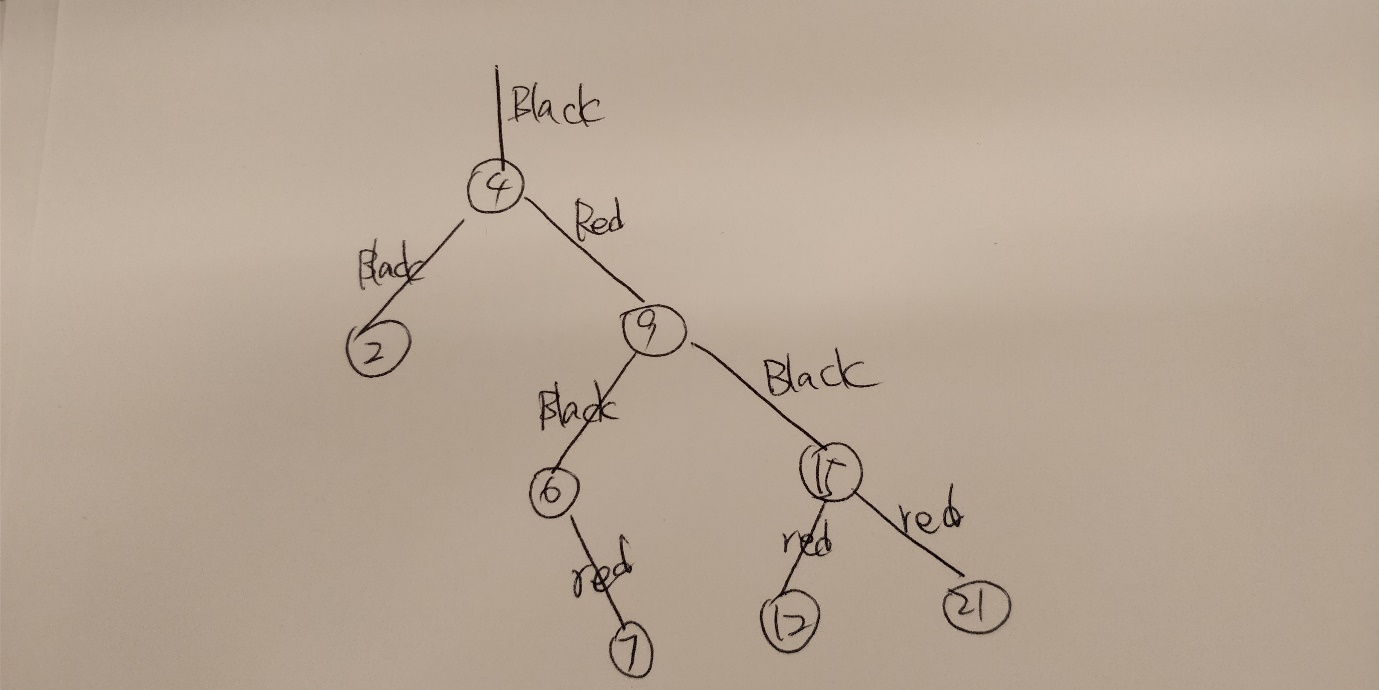
14 is inserted to the right of 12, which is This is the double red with black sibling case, so we perform left rotation followed by a right rotation.

1. In the 2-3 tree given below (i.e., NOT a 2-3-4 tree), execute insert(28), insert(99), and insert(58), in that order, making sure to rebalance after each insertion. Draw the resulting 2-3 tree after executing these operations.

Always insert at leaf node, increase height only when the nodes are full  
  




1. Given the following 2-3-4 tree, draw an equivalent Red-Black tree.  
     
    



**Graphs:**

B

1. A graph G is an undirected graph with n vertices and each vertex has exactly degree d. Assume n is odd. What can you say about d?
   1. d must be odd
   2. d must be even
   3. d can be odd or even

The total number of edges is n\*d/2 and it’s an integer. So n\*d is even, since n is odd, d has to be even to make n\*d even.

B

1. Given a connected graph of n vertices and m edges, with no self-loops or parallel edges, the minimum and maximum number of edges of the graph is:
   1. n <= m <= 2 n
   2. n - 1 <= m <= n (n - 1) / 2
   3. n <= m <= n2
   4. None of the above

Minimum:(if each vertex has exactly 1 degree) n-1

Maximum:(if each vertex has n-1 degrees) n(n-1)/2

A

1. Let G be an undirected graph with n vertices and m edges. If m >= n, then G has a cycle.
   1. Always true
   2. Sometimes true
   3. Never true

If m >=n, then at least 1 vertex has access to all others

C

1. After running DFS on a graph G, how many of the following statements about the edge e=(u -> v) are correct?
   * e is a tree edge if and only if du < dv < fv < fu and while exploring the edge e, vertex v has not been visited yet.
   * e is a forward edge if and only if du < dv < fv < fu and while exploring the edge e, vertex v has been visited before.
   * e is a back edge if and only if dv < du < fu < fv.
   * e is a cross edge if and only if dv < du < fu < fv.

dx is a *discovery* *time* of vertex x, when it is first processed and fx is a *finish* time of vertex x, when all of its descendants are finished.

1. 1
2. 2
3. 3
4. 4

A: correct, u should be discovered first and finished first with no back, forward and cross conditions

B: should be du < dv < fv < fu, correct

C: should be dv < du < fu < fv, correct

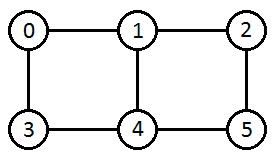
D: should be dv < fv < du < fu, not correct

B

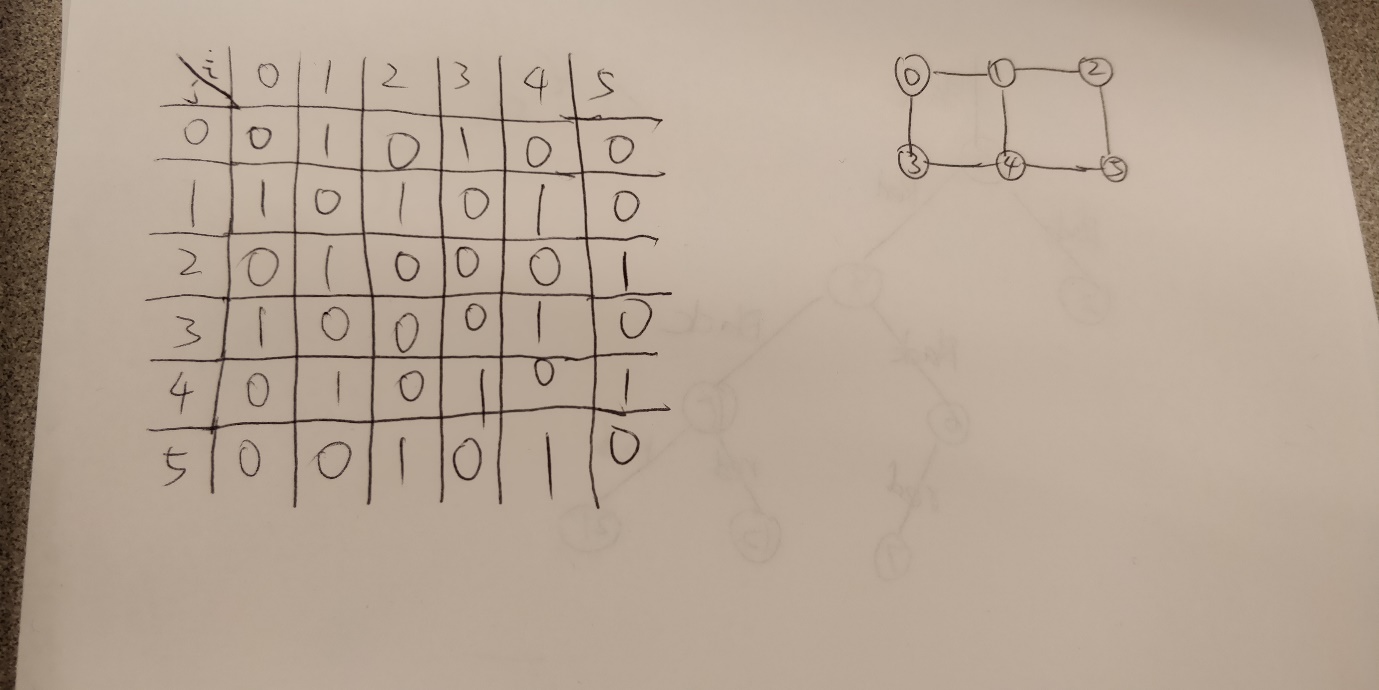
1. A unicycle graph is a connected graph with exactly one cycle. For a given undirected graph G with n vertices and m edges, what is the fastest algorithm to find out if G is unicycle or not?
   1. Θ(n)
   2. Θ(m+n)
   3. Θ(n2)
   4. Θ(mn)

DFS is fastest. It labels n vertices twice and labels m edges twice. so Θ(m+n)

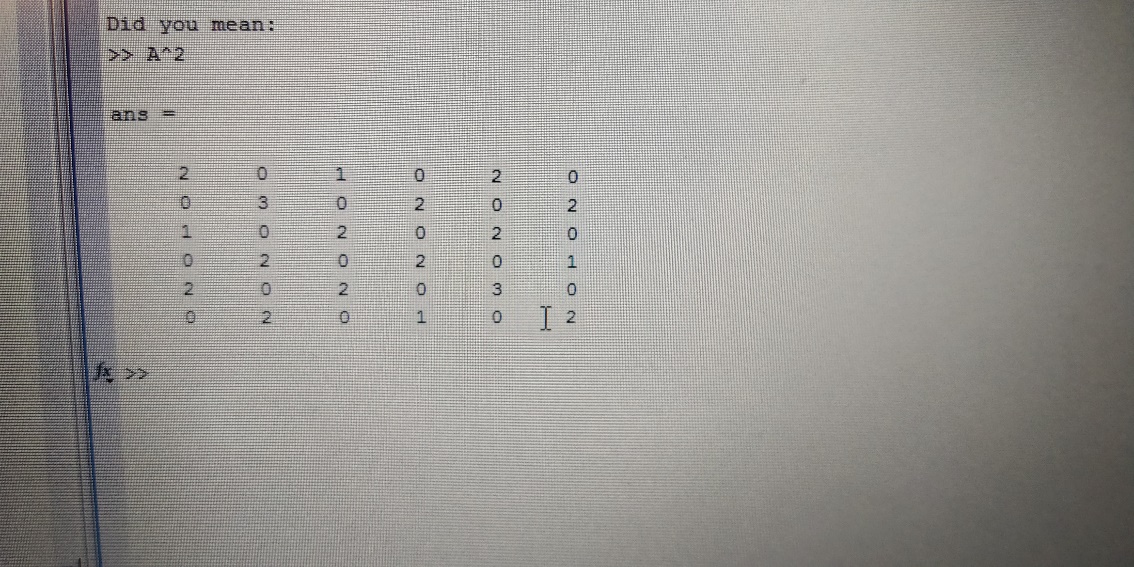
For questions 18, 19, and 20, consider the following graph G:



1. Write down the adjacency matrix for G.



1. Compute the square of the adjacency matrix of G and explain what property of the graph each entry aij shows.

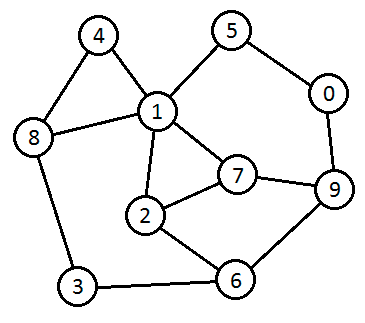


The entries aij shows the number of 2-length paths between the nodes i and j

1. Consider the k-th power of the adjacency matrix (you do not have to compute it). What property of the graph does each entry aij show?

The entries aij shows the number of K-length paths between the nodes i and j

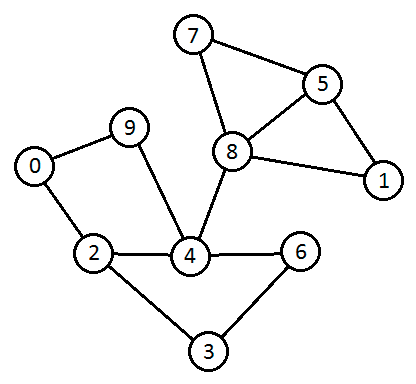
B

1. Which sequence corresponds to a BFS on the following graph, starting from vertex 7?  
    
   1. 7, 1, 2, 9, 0, 4, 5, 6, 8, 3
   2. 7, 9, 1, 2, 6, 0, 5, 8, 4, 3
   3. 7, 2, 1, 9, 6, 5, 4, 8, 3, 0
   4. 7, 1, 2, 9, 8, 4, 0, 6, 3, 5

Furthest is 3, so c and d are not correct.

A doesn’t have the correct scanning process of second children because we are scanning 1’s children first and 0 is not.

A

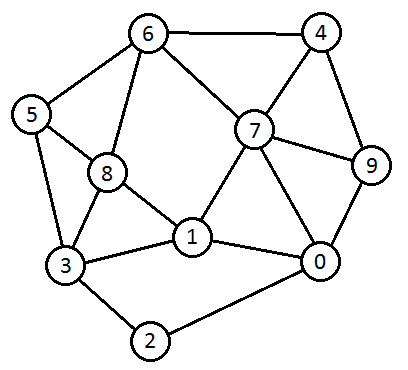
1. Which sequence does NOT correspond to a DFS on the following graph, starting from vertex 3?
   1. 3, 6, 4, 8, 5, 7, 9, 0, 2, 1
   2. 3, 6, 4, 9, 0, 2, 8, 1, 5, 7
   3. 3, 2, 4, 6, 8, 1, 5, 7, 9, 0
   4. 3, 2, 0, 9, 4, 6, 8, 7, 5, 1

B is traversing 3 6 4 with paths traversing 9 0 2 and 8 1 5 7

C is traversing 3 2 4 with paths traversing 6 and 8 1 5 7 and 9 0

D is traversing 3 2 0 9 4 with paths traversing 6 and 8 7 5 1

A D

1. Which vertices, if individually removed, would cause the following graph to no longer be biconnected? Choose all that apply.
   1. 0
   2. 1
   3. 2
   4. 3
   5. 4
   6. 5
   7. 6
   8. 7
   9. 8
   10. 9

Node 2 has 2 degrees. By removing its 2 adjacent nodes, it will be disconnected. So, removing 0 or 3 will make it not biconnected.