Suppose you are given an array s[1..n] and a procedure reverse (s, i, j) which is the reverse-order of elements in s between positions i and j (both inclusive). What does the following sequence do, where 1≤x<n: reverse (s, 1, x);

\*

1 point

reverse (s, x+1, x);

reverse (s, 1, n);

Rotates s left by x positions

Leaves s unchanged

What is the recursive traversing of Pre-order traversal  
\*

1 point

a) traverse the left subtree, visit the root node and traverse the right sub-tree

b) visit the root node, traverse the left sub-tree, and traverse the right sub-tree

c) traverse the left sub-tree, traverse the right sub-tree, and visit the root node

d) None of the above

A binary search tree T contains n distinct elements. What is the time complexity of picking an element in T that is smaller than the maximum element in T?

\*

1 point

Θ(nlogn)

Θ(n)

Θ(logn)

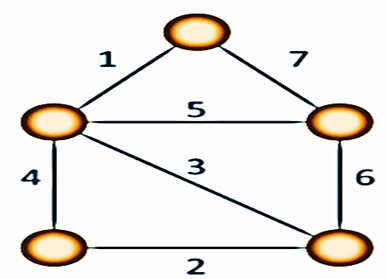
Θ(1)

Consider the following undirected graph with edge weight as shown:

The minimum-weight spanning trees of the graph is ----

\*

1 point



10

11

12

11.5

The concatenation of two lists is to be performed in O(1) time. Which of the following implementations of a list should be used?

\*

1 point

a. Singly linked list

b. Doubly linked list

c. Circular doubly linked list

d. Array implementation of lists

What is the use of Dijkstra’s algorithm?

\*

1 point

Job sequencing

Find the minimum spanning tree

Single source shortest path

None of these

An advantage of chained hash table (external hashing) over the open addressing scheme is

\*

1 point

a. Worst case complexity of search operations is less

b. Space used is less

c. Deletion is easier

d. None of the above

In-order, pre-order and post-order can be applied to  
  
\*

1 point

any trees

only binary trees

any trees other than binary trees

None of the above

*The Floyd-Warshall algorithm for all-pair shortest paths computation is based on*

\*

1 point

a. Greedy paradigm

b. Divide-and-Conquerparadigm.

c. Dynamic Programing paradigm.

d. neither Greedy nor Divide-and-Conquer nor Dynamic Programming paradigm

In a complete k-ary tree, every internal node has exactly k children or no child. The number of leaves in such a tree with n internal nodes is:

\*

1 point

nk

(n-1)k+1

n(k-1)+1

n(k-1)

The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree (the height is the maximum distance of a leaf node from the root)?

\*

1 point

2

3

4

6

What is the worst-case performance of Selection sort algorithm?

\*

1 point

O(log n)

O(n\* n)

O(n)

O(n log n)

A binary search tree T contains n distinct elements. What is the time complexity of picking an element in T that is smaller than the maximum element in T?

\*

1 point

Ө(n log n)

Ө(n)

Ө(log n)

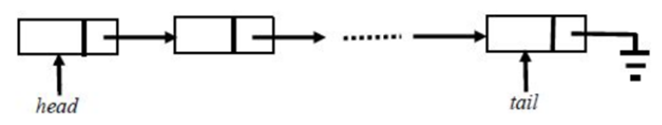
Ө(1)

A queue is implemented using a non-circular singly linked list. The queue has a head pointer and a tail pointer, as shown in the figure. Let n denote the number of nodes in the queue. Let 'enqueue' be implemented by inserting a new node at the head, and 'dequeue' be implemented by deletion of a node from the tail.

Which one of the following is the time complexity of the most time-efficient implementation of 'enqueue' and 'dequeue, respectively, for this data structure?

\*

1 point



a) Ө(1), Ө(1)

b) Ө(1), Ө(n)

c) Ө(n), Ө(1)

d) Ө(n), Ө(n)

Suppose each set is represented as a linked list with elements in arbitrary order. Which of the operations among union, intersection, membership,and cardinality will be the slowest?

\*

1 point

Union only

Intersection, membership

Membership, cardinality

Union, intersection

What is the worst-case number of arithmetic operations performed by recursive binary search on a sorted array of size n?

\*

1 point

Ө(√n)

Ө(log(n))

Ө(n2)

Ө(n)

The minimum number of fields with each node of doubly linked list is

\*

1 point

1

2

3

4

We have a binary heap on n elements and wish to insert n more elements (not necessarily one after another) into this heap. The total time required for this is

\*

1 point

a. Θ (logn)

b. Θ (n)

c. Θ (nlogn)

d. Θ(n 2 )

Consider an implementation of the unsorted single linked list. Suppose it has its representation with a head and a tail pointer (i.e. pointers to the first and last nodes of the linked list). Given the representation, which of the following operation can not be implemented in O(1) time?

\*

1 point

Insertion at the front of the linked list.

Insertion at the end of the linked list.

Deletion of the front node of the linked list.

Deletion of the last node of the linked list.

What this code is doing in a Binary search tree?  
void do\_job(BST node){

If(node!=NULL)

{

            do\_job (node.left());

            do\_job (node.right());

            cout<<node.data;

}

}

\*

1 point

a) Traversing post-order

b) Traversing pre-order

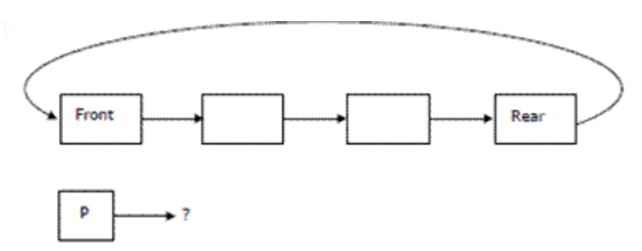
c) Traversing in-order

d) Finding the dept

A circularly linked list is used to represent a Queue. A single variable p is used to access the Queue. To which node should p point such that both the operations enQueue and deQueue can be performed in constant time?

\*

1 point



a) Rear node

b) Front node

c) Not possible with a single pointer

d) Node next to front

What is the best case complexity of quick sort?

\*

1 point

Ω(n)

Ɵ(logn)

Ω(n(log n))

Ω(log n)

Given a binary-max heap. The elements are stored in an arrays as 25,14,16,13,10,8,12. What is the content of the array after two delete operations?

\*

1 point

a. 14,13,8,12,10

b. 14,12,13,10,8

c. 14,13,12,8,10

d. 14,13,12,10,8

A single array A[1..MAXSIZE] is used to implement two stacks, The two stacks grow from opposite ends of the array. Variables top1 and top2 (top1 < top2) point to the location of the topmost element in each of the stacks, If the space is to be used efficiently, the condition for "stack full" is

\*

1 point

(top 1 = MAXSIZE/2) AND (top 2 = MAXSIZE/2 + 1)

top 1 + top 2 = MAXSIZE

(top 1 = MAXSIZE/2) or (top 2 = MAXSIZE)

top 1 = top 2 – 1

What sorting algorithms have equal best case and worst case time complexity?

\*

1 point

heap and selection sort

insertion sort & merge sort

merge sort and heap sort

None of these

*Point mutations* of strings str1 into str2 are  
  
\*

1 point

change a letter

insert a letter or

delete a letter

Any one of the above

Which of the following types of Linked List support forward and backward traversal?

\*

1 point

A. Singly Linked List

B. Doubly Linked List

C. Circular Singly Linked List

D. All of these

Which of the following are related to stack?  
\*

1 point

push

pop

LIFO

All of the above

Merge sort uses \_\_\_\_\_\_\_\_\_\_\_\_\_ strategy

\*

1 point

backtracking

heuristic

greedy

divide and conquer

What is the time complexity of build Heap operation. Build Heap is used to build a max(or min) binary heap from a given array. Build Heap is used in Heap Sort as a first step for sorting  
\*

1 point

a. O(nlogn)

b. O(n2)

c. O(logn)

d. O(n)

How much time is required by Prim’s algorithm of Graph(G) & n is the number of vertices?  
\*

1 point

O(n)

O(n^2)

O(log n)

O(n long n)

The number of rotations required to insert a sequence of elements 9, 6, 5, 8, 7, 10 into an empty AVL tree is?

\*

1 point

0

1

2

3

Convert the following infix expression into their Postfix form  
(X^Y)/(A\*B)  
  
\*

1 point

/ ^ XY \* A B

XY ^ AB \* /

X ^ Y AB \* /

None of the above

Queue can be used to implement

\*

1 point

radix sort

quick sort

recursion

depth first search

Which of the following condition is sufficient to detect cycle in a directed graph?

\*

1 point

There is an edge from currently being visited node to an ancestor of currently visited node in DFS forest.

There is an edge from currently being visited node to an already visited node.

Every node is seen twice in DFS.

None of the above

Floyd-Warshall algorithm utilizes \_\_\_\_\_\_\_\_\_\_ to solve the all-pairs shortest paths problem on a directed graph in \_\_\_\_\_\_\_\_\_\_ time.

\*

1 point

a. Greedy algorithm, θ (V3)

b. Greedy algorithm, θ (V2log n)

c. Dynamic Programming, θ (V3)

d. Dynamic Programming, θ (V2 log n)

What is recurrence for worst case of QuickSort and what is the time complexity in Worst case?

\*

1 point

a. Recurrence is T(n) = T(n-1) + O(n) and time complexity is O(n2)

b. Recurrence is T(n) = T(n-2) + O(n) and time complexity is O(n2)

c. Recurrence is T(n) = 2T(n/2) + O(n) and time complexity is O(nlog n)

d. Recurrence is T(n) = T(n/10)+T(9n/10) + O(n) and time complexity is O(nlog n)

In \_\_\_\_\_\_\_\_\_the exploration of node is suspended as soon as new unexplored node is reached.  
\*

1 point

BFS

DFS

Prims algorithm

Kruskal’s algorithm

A Stack structure would require  
\*

1 point

head pointer to remove an existing node

tail pointer to add to a new node

both (a) and (b)

None of the above

In a doubly linked list, the number of pointers affected for an insertion operation will be

\*

1 point

4

0

1

None of the above