MCQS Data structure

10. What is the worst case time complexity to search an element from a singly linked list of length n.

1. O(log n)

2. O(n log n)

3. O(n) //correct

4. O(1)

9. A linked List where there are no NULL links is :

1. Singly linear linked list

2. Doubly linear linked list

3. Circular linked list // correct

4. All of the above

8. Below one is add\_first() functionality code for empty Doubly CircularLinked list. Find the correct statement to replace ????

head = newnode;

tail = newnode;

tail ->next = head;

????

1. tail->next->next = tail

2. head->prev = tail; // correct

3. tail->prev = head;

4. head->prev = NULL;

7. consider following implementation is for the unsorted doubly linked list. And consider with only head pointer. Time complexity of the which given implementation is o(1)?

1. insertion the element and delete the node in unsorted linked list.

2. insertion the element in front and delete the front node. // correct

3. insertion the element at end.

4. deletion the middle node.

6. What is the output of the following code for start pointing to first node of the given list? 1->2->3->4->5->6

void Linkedlist(struct node\* start)

{

if(start == NULL)

return;

printf("%d",start->data);

if(start->next != NULL)

Linkedlist(start->next->next);

printf("%d",start->data);

}

1. 1 4 6 6 4 1

2. 1 3 5 1 3 5

3. 1 2 3 5

4. 1 3 5 5 3 1 // coreect

5. Which of the following operations is performed more efficiently by doubly linked list than by singly linked list?

Answers

1. Deleting a node whose location in given // correct

2. Searching of an unsorted list for a given item

3. Inverting a node after the node with given location

4. Traversing a list to process each node

1. what is wrong about singly linked list?

1. Singly linked list is a collection of nodes linked together in a sequential way where each node of singly linked list contains a data field

2. singly list is an address field which contains the reference of the next node.

3. Singly linked list can contain multiple data fields but should contain at least single address field pointing to its connected next node.

4. None Of Above // correct

2. What would be the asymptotic time complexity to add a node at the end of singly linked list, if the pointer is initially pointing to the head of the list?

1. O(1)

2. O(n)

3. θ(n) // correct

4. Θ(1)

3. which statement is True about circular linked list?

1. Entire list can be traversed from any node.

2. Circular lists are the required data structure when we want a list to be accessed in a circle or loop.

3. Despite of being singly circular linked list we can easily traverse to its previous node, which is not possible in singly linked list.

4. All of Above // correct

4. Which of the following is false about a doubly linked list?

1. We can navigate in both the directions

2. It requires more space than a singly linked list

3. The insertion and deletion of a node take a bit longer

4. Implementing a doubly linked list is easier than singly linked list // correct