Questions for Question Bank

21CSC201J: Data Structures and Algorithms

Unit – II: Single Linked List

MCQ

1 What is the time complexity to count the number of elements in the linked list?
a) O(1)
b) O(n)
c) O(logn)
d) $O(n^2)$

2.What is the functionality of the following code?

```
public void function(Node node)
{
        if(size == 0)
                head = node;
        else
        {
                Node temp,cur;
                for(cur = head; (temp = cur.getNext())!=null; cur = temp);
                cur.setNext(node);
        }
        size++;
}
```

a) Inserting a node at the beginning of the list
b) Deleting a node at the beginning of the list
c) Inserting a node at the end of the list
d) Deleting a node at the end of the list

3.Which of these is not an application of a linked list?
a) To implement file systems
b) For separate chaining in hash-tables
c) To implement non-binary trees
d) Random Access of elements

4.How do you insert an element at the beginning of the list?

a)

```
public void insertBegin(Node node)
{
        node.setNext(head);
        head = node;
        size++;
}
```

b)

```
public void insertBegin(Node node)
{
```

```
        head = node;
        node.setNext(head);
        size++;
}
```
c)

```
public void insertBegin(Node node)
{
        Node temp = head.getNext()
        node.setNext(temp);
        head = node;
        size++;
}
```
d)

```
public void insertBegin(Node node)
{
        Node temp = head.getNext()
        node.setNext(temp);
        node = head;
        size++;
}
```

5.What is the functionality of the following piece of code?

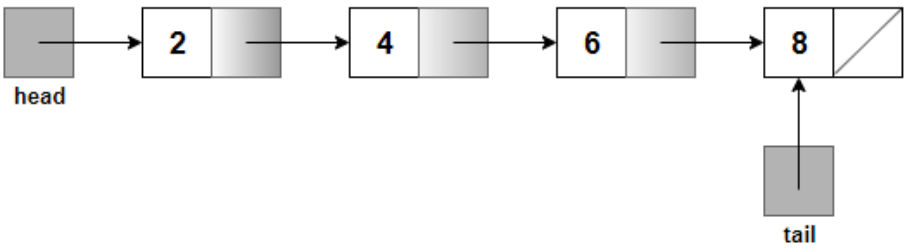
```
public int function(int data)
{
        Node temp = head;
        int var = 0;
        while(temp != null)
        {
                if(temp.getData() == data)
                {
                        return var;
                }
                var = var+1;
                temp = temp.getNext();
        }
        return Integer.MIN_VALUE;
}
```
a) Find and delete a given element in the list
b) Find and return the given element in the list
c) Find and return the position of the given element in the list
d) Find and insert a new element in the list

Understanding question [Blooms level two]

1.How will you represent a linked list in a graphical view?

A linked list may be defined as a data structure in which each element is a separate object. Linked list elements are not kept at the contiguous location. The pointers are used to link the

elements of the Linked List. Each node available in a list is made up of two items- **the data itself** and **a reference (also known as a link) to the next node** in the sequence. The last node includes a reference to null. The starting point into a linked list is known as the head of the list. It should be noted that the head is a reference to the first node, not a separate node. The head is considered as a null reference if the list is empty.



Linked List Representation

2.Mention the steps to insert data at the starting of a singly linked list?

Steps to insert data at the starting of a singly linked list include,

Create a new node

Insert new node by allocating the head pointer to the new node next pointer

Updating the head pointer to the point the new node.

```
Node *head;

void InsertNodeAtFront(int data)

{

/* 1. create the new node*/

Node *temp = new Node;

temp->data = data;

/* 2. insert it at the first position*/

temp->next = head;

/* 3. update the head to point to this new node*/

head = temp;

}
```

3.Mention what is the biggest advantage ofsingle linked lists?

The biggest benefit of single linked lists is that you do not specify a fixed size for your list. The more elements you add to the chain, the bigger the chain gets.

4.How should the first node in a singly linked list be deleted?

To delete first node from singly linked list

Store Current Start in Another Temporary Pointer

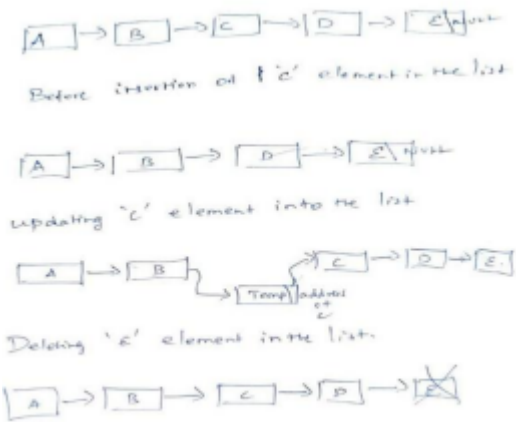
Move Start Pointer One position Ahead

Delete temp i.e Previous Starting Node as we have Updated Version of Start Pointer

Scenario Based

1. David preparing the list of things needed for his college tour. He added 5 items in the list (Item A, Item B, Item C, Item D, and Item E). After that, he realized that one more item is also very important for his tour trip. So, he is trying add the new item after Item C in the list and also, he removed Item E from the list. Try to help him to do these operations using suitable data structure.

With the help of it can be implemented Single linked list :-




```
struct node *newNode;
newNode = malloc(sizeof(struct node));
newNode->data = 4;
newNode->next = head;
head = newNode;
```

For insertion

```
struct node *newNode;
newNode = malloc(sizeof(struct node));
newNode->data = 4;
newNode->next = NULL;
struct node *temp = head;
while(temp->next != NULL){
```

```
  temp = temp->next;
 }
temp->next = newNode;
For insertion in middle
struct node *newNode;
newNode = malloc(sizeof(struct node));
newNode->data = 4;
struct node *temp = head;
for(inti=2; i< position; i++)
{
if(temp->next != NULL)
{
temp = temp->next;
}
}
newNode->next = temp->next;
temp->next = newNode;
```

2.Consider a goods train, which initially has an engine as a header. Later compartments are linked with engines one by one in series. The compartments can be attached and detached with engine or in between the compartments. Implement a data structure for the following operations in the goods train

1. Attaching a compartment in between the compartments
2. Detaching a compartment in between the compartments.

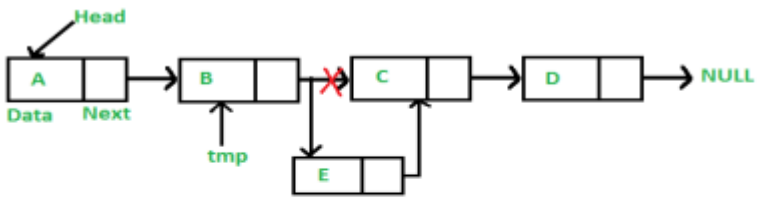
Attaching Compartment -5 marks

Detaching compartment - 5 Marks

The suitable data structure for the above problem is linked list

1: Attaching a Compartment in between the compartments'

Insert at the Middle (A,B,C,D are the compartments)



- Allocate memory and store data for new node
- Traverse to node just before the required position of new node

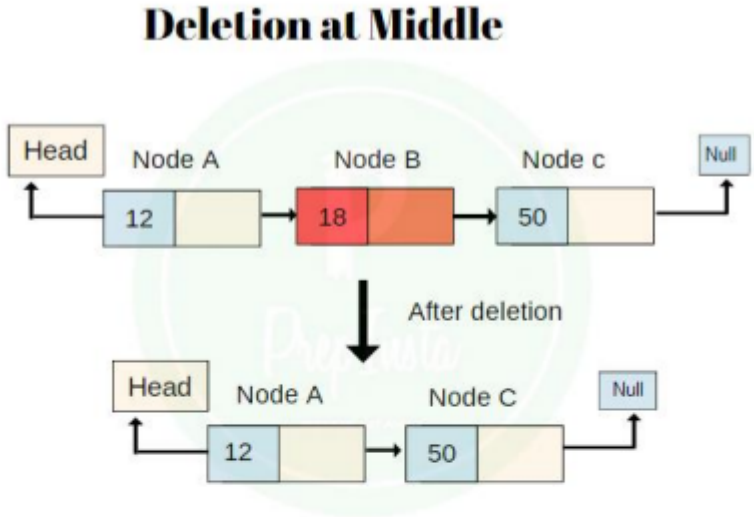
- Change next pointers to include new node in between

```
struct node *newNode;
newNode = malloc(sizeof(struct node));
newNode->data = 4;

struct node *temp = head;

for(int i=2; i < position; i++) {
  if(temp->next != NULL) {
    temp = temp->next;
  }
}
newNode->next = temp->next;
temp->next = newNode;
```

Delete from middle



- Traverse to element before the element to be deleted
- Change next pointers to exclude the node from the chain

```
for(int i=2; i< position; i++) {
  if(temp->next!=NULL) {
    temp = temp->next;
  }
}

temp->next = temp->next->next;
```