Table of Contents

Contributors

User	, Answers
Shishir Roy	25, 3
shadymademe	19, 8
Sai Krishna Lolla	8, 2
GO Classes for GATE CSE	7, 1
NAMAN SHARMA	5, 1
sidd_07	2, 1
Prateek Jha	1, 2

User	?Added
GO Classes for GATE CSE	20

User	Done
Shishir Roy	3
Sachin Mittal	1
Pranav Purkar	1
Lakshman Patel	1
Sai Krishna Lolla	1
NAMAN SHARMA	1

1 Programming and DS: DS (20)

1.1 Linked List (20) top

1.1.1 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 1 top

Consider a sorted circular doubly-linked list where the head element points to the smallest element in the list. Which of the following(s) is/are true?

A. Asymptotic time complexity of finding the smallest element in the list is $O(1)$
B. Asymptotic time complexity of finding the largest element in the list is $O(1)$
C. Asymptotic time complexity of determining whether a given element e appears in the list is $O(\log n)$
D. Asymptotic time complexity of deleting a given element e in the list (not including the cost of finding it) is $O(1)$

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 1-mark

Answer key

1.1.2 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 10 top

Consider the function insertBeginning() below which inserts a node in the beginning of a doubly linked list. Choose the correct option to fill commented line in below code such that insertBeginning() works as stated.

Assuming that "struct node" is a structure with three usual fields for a doubly linked list (prev, next, and val), where all the fields work as expected by their name i.e. prev points to the previous node and so on.

```
struct node * insertBeginning(struct node *head, int val){
  struct node *newNode = malloc(sizeof( struct node ));
  newNode->value = val;
  // your code goes here
  return head;
}
```

A.

```
newNode->prev = NULL;
newNode->next = head;
head->prev =newNode;
head =newNode;
```

B.

```
newNode->prev = head;
newNode->next = NULL;
head->prev =newNode;
head =newNode;
```

C.

```
newNode->prev = head;
newNode->next = NULL;
head->next =newNode;
head =newNode;
```

D.

```
newNode->prev = NULL;
newNode->next = head;
head->prev =newNode;
head =newNode->next;
```

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 1-mark

Answer key

1.1.3 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 11 top

Consider the function removeElements(int marker), which intends to remove all nodes with value equal to marker from the link list, leaving the rest intact. If no node with value marker exists in the list, then the list should left unchanged.

For example: calling removeElements(head, 12) on the linked list : $5 \rightarrow 12 \rightarrow 6 \rightarrow 3 \rightarrow 12 \rightarrow 12 \rightarrow 9$ should result in : $5 \rightarrow 6 \rightarrow 3 \rightarrow 9$.

```
void removeElements(Node * head, int marker) {
```

```
Node * current = head;
while (current != NULL) {
    if (current -> next != NULL && current -> next -> data == marker) {
        Node * newNext = current -> next -> next;
        while (newNext != NULL && newNext -> data == marker) {
            newNext = newNext -> next;
        }
        current -> next = newNext;
    }
    current = current -> next;
}
}
```

Node is singly linked list type typedef structure and head always points to the first node of linked list.
Which of the option(s) is/are correct about removeElements() if we call this function from the main() function of some C program.

A. removeElements(head, 12) on linked list : $5 \rightarrow 12 \rightarrow 6 \rightarrow 3 \rightarrow 12 \rightarrow 12 \rightarrow 9$ would result in : $5 \rightarrow 6 \rightarrow 3 \rightarrow 9$.
B. removeElements(head, 12) on linked list : $12 \rightarrow 6 \rightarrow 3 \rightarrow 12 \rightarrow 12 \rightarrow 9$ would result in : $6 \rightarrow 3 \rightarrow 9$.
C. removeElements(head, 12) on linked list : $6 \rightarrow 3 \rightarrow 12 \rightarrow 12$ would result in : $6 \rightarrow 3$.
D. removeElements(head, 12) on linked list : $12 \rightarrow 6 \rightarrow 3 \rightarrow 12 \rightarrow 12$ would result in : $6 \rightarrow 3$.

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 2-marks

Answer key

1.1.4 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 12 top

Consider a function ReverseList() which has following protype -

```
Node *ReverseList(Node *p)
```

Function reverse the nodes in a given linked list.

That is, head = ReverseList(head); will take a list held by head, reverse it and put back to head.

```
Node * ReverseList(Node * p) {
    if (!p) return NULL;
    if (p -> next) {
        Node * q = ReverseList(p -> next);
        p -> next -> next = p;
        p -> next = 0;
        return ______; //Line X
    }
    return ______; //Line Y
}
```

Fill in the blanks for Line X and Line Y. If we say Line X is filled by p then we consider Line X as **return p;** after filling the blank.
Line X and Line Y should be filled by which values respectively such that the function correctly reverses the given linked list.

A. p and $p \rightarrow$ next B. p and q C. $q \rightarrow$ next and p D. q and p

goclasses2024_wq20 goclasses data-structures linked-list 2-marks

Answer key

1.1.5 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 13 top

Consider the following function recursive_remove().

```
Node* recursive_remove(Node* head, int x)
{
    if (head == NULL)
        return NULL;
    if (head->value == x)
    {
        Node* tmp = head->next;
        free(head);
        return recursive_remove(tmp, x);
    }
    else
    {
        head->next = recursive_remove(head->next, x);
        return head;
    }
}
```

If we call the function recursive_remove(head2) on the following linked list then $1 \rightarrow 2 \rightarrow 2 \rightarrow 8 \rightarrow 6 \rightarrow 2 \rightarrow 2$

A. Final LinkedList will be $1,2,8,6,2,2$
B. Final LinkedList will be $1,8,6$
C. Final LinkedList will be $1,8,6,2,2$
D. None of the above

goclasses2024_wq20 goclasses data-structures linked-list 2-marks

Answer key

1.1.6 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 14 top

Consider the function mystery which takes the head of a singly linked list as an argument.

```
int mystery(Node* head)
{
   if (head == NULL) return 0;
   else return head->data - mystery(head->next);
}
```

If the linked list contains integers from 1 to n in the order, where the head is pointing to1.

Which of the following is/are true about the function call mystery(head)?

A. It returns -50 if n is 99
B. It returns 50 if n is 99
C. It returns -50 if n is 100
D. It returns 50 if n is 100

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 2-marks

Answer key

1.1.7 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 15 top

Consider the following function fun() that takes the head of a linked list.

```
struct node {
   int value;
   struct node *next;
};
typedef struct node Node;

int fun(Node *head){
   if(head== NULL) return 1;
   Node *p,*q;
   p = head;
   q = p->next;
   while(q!=NULL && q!=p){
       q = q->next;
   if(q==NULL) return 1;
   q = q->next;
   p = p->next;
   }
return (q==NULL);
}
```

We say, a linked list has a loop if the last node of linked list points to some node of linked list and does not point to NULL.

What does the above function do?

A. Returns 0 is there is loop in linked list
B. Returns 1 is there is loop in linked list
C. Returns 0 is length of the linked list is even
D. Function may go to infinite loop if there is a loop in linked list

goclasses2024_wq20 goclasses data-structures linked-list 2-marks

Answer key

1.1.8 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 16 top

Consider an unsorted singly linked list.

Suppose it has as its representation with a head and tail pointer (i.e., pointers to the first and last nodes in the list). Given that representation, which of the following operations could be implemented in $O(1)$ time?

I. Insert item at the front of the list
II. Insert item at the rear of the list
III. Delete the front item from the list

IV. Delete rear item from the list

A. I and II B. I and III C. I, II, and III D. I, II, and IV

goclasses2024_wq20 goclasses data-structures linked-list 2-marks

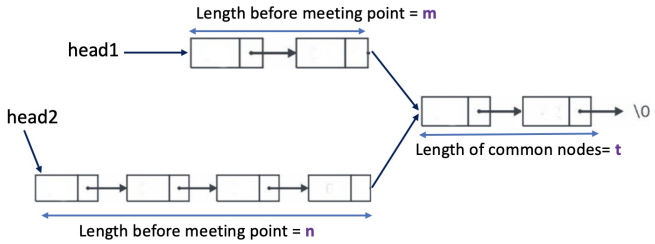
Answer key

1.1.9 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 17 top

Suppose there are two singly linked lists both of which intersect at some point and become a single linked list. See figure for visual understanding.

The head or start pointers of both the lists are known, but the intersecting node is not known. Also, the number of nodes in each of the list before they intersect are unknown and both list may have it different i.e. List1 may have m nodes before it reaches intersection point and List2 might have n nodes before it reaches intersection point where $m(\geq 2)$ and $n(\geq 2)$ may be

- $m = n$,
- $m < n$ or
- $m > n$



Consider the below algorithm which tries to find out the intersecting point of both linked lists -
Here M and N are total lengths of linked list 1 and 2 respectively. That is, $\mathrm{M} = m + t$ and $\mathrm{N} = n + t$.

1. Traverse the two linked lists to find M and N.
2. Get back to the heads, then traverse $|\mathrm{M} - \mathrm{N}|$ nodes on the long list.
3. Now start walking in the first linked list too and compare the nodes until you find the common ones. I.e. if $\mathrm{M} = 4$ and $\mathrm{N} = 6$ then we will compare 1st node address in linked list 1 with 3rd node address in linked list 2, then 2nd node address in linked list 1 and 4th node address in linked list 2, and so on.

Which of the following is/are correct about this algorithm?

A. Algorithm will fail to detect the intersection point for $m = n$.
B. Algorithm will fail to detect the intersection point for $m < n$.
C. Algorithm will correctly detect the intersection point for every input and the time complexity of the algorithm is $O(\mathrm{M}+\mathrm{N})$.
D. Algorithm will correctly detect the intersection point for every input and the time complexity of the algorithm is $O(\mathrm{MN})$.

goclasses2024_wq20 goclasses data-structures linked-list 2-marks

Answer key

1.1.10 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 18 top

Which of the following prints the last node of a circular (non empty) linked list?

In all options, head is pointing to the first node of the circular (non empty) linked list.

A.

```
struct node *current = head;
while(current->next!=head)
current=current->next;
printf("%d", current->value);
```

B.

```
struct node *current = head->next;
while(current->next!=head)
current=current->next;
printf("%d", current->value);
```

C.

```
struct node *current = head;
while(current!=head->next)
current=current->next;
printf("%d", current->value);
```

D.

```
struct node *current = head;
while(current->data!=head->data)
current=current->next;
printf("%d", current->value);
```

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 2-marks

Answer key

1.1.11 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 19 top

Consider two programs given below. Consider a non-empty linked list which is either a circular or singly linked list. $P1$ and $P2$ are two programs which try to check if it is circular or not. Let $head$ point to the first node of the linked list.

$P1$ takes the head of the linked list as its argument and $P2$ takes head as the first argument and head->next as the second argument at the first call.

```
int P1(struct node* head)
{
   if (head==null)
      return 1;

   struct node* next = head->next;

   while(next!=null && next !=head)
   next = next->next;
   return (next == head);

}
```

```
int P2(struct node* head, struct node* cur)
{
   if (head==null)
      return 1;

   if(cur==null)
      return 0;

   if(head==cur)
      return 1;

   return P2(head, cur->next);
}
```

Which of the following is/are true?

A. $P1$ returns 1 if given linked list is circular
B. $P2$ returns 1 if given linked list is circular
C. $P1$ returns 1 if given linked list is having even number of nodes
D. $P2$ returns 1 if given linked list is having even number of nodes

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 2-marks

Answer key

1.1.12 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 2 top

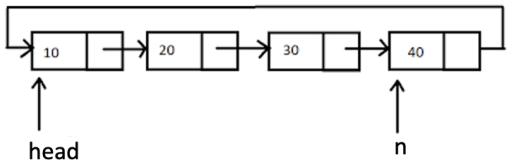
Consider a linked list given in the figure below.

What is the value of $n \rightarrow next \rightarrow next \rightarrow next \rightarrow data$?

Where n and head are pointers to the following struct type and intitallised as per the figure shown.

```
struct node{
```

```
    int data;
    struct node *next;
};
```



A. 10 B. 20 C. 30 D. 40

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.13 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 20 top

Consider the following function **printNthNode()**.

```
void printNthNode(Node *head){
    Node *p, *q;
    p = q= head;
    while(q){
        p = p->next;
        q = q->next ? q->next->next : NULL;
    }
    printf("%d", p->data);
}
```

We pass the head of the singly linked list to the above function. Which of the option(s) is/are correct about a linked list of length n $(n \geq 2)$?

The length of a linked list is the number of nodes in it.

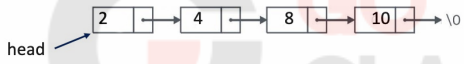
A. Function prints $n/2$ node value if n is even
B. Function prints $n/2 + 1$ node value if n is even
C. Function prints $(n-1)/2 + 1$ node value if n is odd
D. Function prints $(n+1)/2 + 1$ node value if n is odd

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 2-marks

Answer key

1.1.14 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 3 top

Consider the following singly linked list where the head points to the first node of the linked list.



If we execute the following code on the above-linked list then what will be the final linked list?

```
Node *p, *addNode;
p = head->next;
addNode = createNode(6);
addNode->next = p->next;
p->next = addNode;
```

Here **createNode()** is a function that takes an integer value and creates a node using **malloc**, sets the data field to an integer value, and the next field to **NULL**.

A. $2 \rightarrow 4 \rightarrow 6 \rightarrow 8 \rightarrow 10$
B. $2 \rightarrow 4 \rightarrow 8 \rightarrow 6 \rightarrow 10$
C. $2 \rightarrow 4 \rightarrow 8 \rightarrow 10$
D. $2 \rightarrow 6 \rightarrow 4 \rightarrow 8 \rightarrow 10$

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.15 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 4 top

The concatenation of 2 lists can be performed $O(1)$ time.

Following are the constraints -

- Only Pointer to head is given
- Swap of node values is not allowed
- The order should be preserved in concatenation. I.e. first list end node should point to the second list first node.

Which of the following implementations of the list should be used?

A. Singly Linked List
B. Doubly Linked List
C. Circular Linked List
D. Circular Doubly Linked List

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.16 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 5 top

What is the output of the following function for the head pointing to the first node of the following linked list?

$$1 \rightarrow 5 \rightarrow 9 \rightarrow 4 \rightarrow 7 \rightarrow 6$$

```
void myFunction(struct node* head)
{
   if(head == NULL)
      return;
   printf("%d ", head->data);
   if(head->next != NULL )
      myFunction(head->next->next);
   printf("%d ", head->data);
}
```

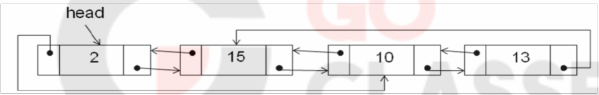
A. $1\ 9\ 7\ 7\ 9\ 1$
B. $1\ 9\ 7$
C. $1\ 9\ 7\ 1\ 9\ 7$
D. $1\ 9\ 7\ 6\ 4\ 5$

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.17 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 6 top

Consider the following linked list which has nodes of a doubly linked list type.



If we execute the following lines of code on a given linked list then what will be the output?

```
head->prev->next->next = head->next->next;
head = head->prev->prev;
head->prev = head;
printf("%d", head->data);
```

A. 2
B. 15
C. 10
D. 13

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.18 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 7 top

Suppose we have a Node data structure declared as follows:

```
struct Node
{
   int item;
   Node *next;
};
```

What value does this function return, assuming that its argument is a properly formed singly linked list with a null pointer in the next field of the last Node?

```
int func(Node * arg) {
   if (arg == NULL) {
      return 0;
   }
   else {
      if (arg -> item >0) return 1 + func(arg ->next);
      else
         return func(arg -> next);
   }
}
```

A. Returns the sum of the positive numbers in the list
B. Returns the number of positive numbers in the list
C. Returns the number of negative numbers in the list
D. Does not execute correctly because it never reaches a base case

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

1.1.19 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 8 top

Consider the following program. printlist() is a function that takes the head of a linked list and prints all nodes values separated by comma. Node is typedefed singly linked list type struct.

```
void insert1(Node *head,int data)
{
   Node *NewNode= (Node *)malloc(sizeof(Node));
   NewNode->value=data;
   NewNode->next=head;
   head=NewNode;
}
void insert2(Node **head_ref,int data)
{
   Node *NewNode= (Node *)malloc(sizeof(Node));
   NewNode->value=data;
   NewNode->next=*(head_ref);
   *(head_ref)=NewNode;
}
int main()
{
   /* create a linked list 1->2->3->4->5
and head points to the first node.*/
   insert1(head,9);
   printlist(head); //Line X

   //The list is restored to its initial state

   insert2(&head,9);
   printlist(head); //Line Y
}
```

Which of the following is/are true about the above program?

A. Line X prints $9,1,2,3,4,5$
B. Line Y prints $9,1,2,3,4,5$
C. Line X prints $1,2,3,4,5$
D. Line Y prints $1,2,3,4,5$

goclasses2024_wq20 goclasses data-structures linked-list multiple-selects 1-mark

Answer key

1.1.20 Linked List: GO Classes 2024 | Weekly Quiz 20 | Linked List | Question: 9 top

The intent of the function below is to delete the last node of the list.

```
void removeLast(Node* first) {
   Node *p, *q;
   p = first;
   q = p->next;
   while (q->next != NULL) {
      p = q;
      q = q->next;
   }
   p->next = NULL;
   free(q);
}
```

Which of the following describes the class of all linked lists for which this function works correctly?

A. No linked lists
B. All non-empty linked lists
C. All linked lists with more than one node
D. The empty list and all linked lists with more than one node

goclasses2024_wq20 goclasses data-structures linked-list 1-mark

Answer key

Answer Keys

1.1.1	A;B;D	1.1.2	A	1.1.3	A;C	1.1.4	D	1.1.5	B

1.1.6	B;C
1.1.11	A;B
1.1.16	A

1.1.7	A
1.1.12	C
1.1.17	B

1.1.8	C
1.1.13	B;D
1.1.18	B

1.1.9	C
1.1.14	A
1.1.19	B;C

1.1.10	A;B
1.1.15	D
1.1.20	C