Summary in Graph

Exam Summary (GO Classes CS Test Series 2025 | Data Structures | Subject Wise Test 1)

Qs. Attempted:	**0** 0 + 0	**Correct Marks:**	**0** 0 + 0
Correct Attempts:	**0** 0 + 0	**Penalty Marks:**	**0** 0 + 0
Incorrect Attempts:	**0** 0 + 0	**Resultant Marks:**	**0** 0 + 0

Total Questions:	**30** 15 + 15
Total Marks:	**45** 15 + 30
Exam Duration:	**90 Minutes**
Time Taken:	**0 Minutes**

EXAM RESPONSE | EXAM STATS | FEEDBACK

Technical

Q #1 | Multiple Select Type | Award: 1 | Penalty: 0 | Algorithms

Suppose

- $\mathrm{A} = (\log\ n)^k$
- $\mathrm{B} = n^\epsilon$

Assume that $k \geq 1$ and $\epsilon > 0$

What is the relation $\mathrm{between}$ the asymptotic time complexities of A and B?

A. $\mathrm{A} = \mathrm{O}(\mathrm{B})$
B. $\mathrm{A} = \mathrm{o}(\mathrm{B})$
C. $\mathrm{A} = \Omega(B)$
D. $\mathrm{A} = \omega(B)$

Your Answer: | **Correct Answer: A;B** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #2 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Algorithms

What is the time complexity of the code given?

```
for(i=1;i<=n;i++)
    for(j=1;j<=n;j=j+i)
        printf("GO Classes");
```

A. $\Theta\left(n^2\right)$
B. $\Theta(n \log n)$
C. $\Theta(n \log \log n)$
D. None of these

Your Answer: | **Correct Answer: B** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #3 Multiple Select Type | Award: 1 | Penalty: 0 | DS

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;
```

Your Answer: | **Correct Answer: A** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #4 Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

If the variables are suitably initialized, and if i remains within appropriate bounds, then the following code implements the stack operations Push and Pop when the stack is represented as an array $V[1 \ldots N]$ with an index variable i.

```
Push: begin V[i]:=x; i:=i+1; end
Pop: begin i:=i-1 ; x:=V[i]; end
```

Which of the following gives the correct initialization for this stack implementation?

A. i:=0
B. i:=1
C. i:=N-1
D. i:=N

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #5 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

Suppose that you implement a queue using a null-terminated singly-linked list, maintaining a reference to the item least recently added (the front of the list) but not maintaining a reference to the item most recently added (the end of the list). What are the worst case running times for enqueue and dequeue?

A. constant time for both enqueue and dequeue
B. constant time for enqueue and linear time for dequeue
C. linear time for enqueue and constant time for dequeue
D. linear time for both enqueue and dequeue

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #6 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Algorithms

What will be the time complexity of following pseudo code?

```
s=1
while s <= (log n)*(log n)
    s=3*s;
```

A. $\Theta\left((\log n)^2\right)$
B. $\Theta(\log n)$
C. $\Theta(\log \log n)$
D. $\Theta(\log \log \log n)$

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #7 | Multiple Select Type | Award: 1 | Penalty: 0 | DS

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 $O(1)$

Your Answer: | Correct Answer: A;B;D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #8 | Multiple Select Type | Award: 1 | Penalty: 0 | DS

Which of the following statements about stacks, queues, and linked lists are true?

A. Stacks and queues can be implemented using arrays.
B. A queue allows a user to retrieve nodes using the LIFO principle, in $\mathrm{O}(1)$ time.
C. Suppose we have a circular array holding only integers, and having a capacity (maximum size) of n. If the array is a circular array, then we can store more than n entries in it at any given time because we can make use of the modulus operator.
D. Using a doubly-linked list, it is not possible to keep the nodes sorted.

Your Answer: | **Correct Answer: A** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #9 Multiple Select Type | Award: 1 | Penalty: 0 | DS

Suppose that a client performs an intermixed sequence of (queue) enqueue and dequeue operations. The enqueue operations put the integers 0 through 9 in order onto the queue; the dequeue operations print out the return value.

Which of the following sequence(s) could not occur?

A. $0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9$
B. $4\ 6\ 8\ 7\ 5\ 3\ 2\ 9\ 0\ 1$
C. $2\ 5\ 6\ 7\ 4\ 8\ 9\ 3\ 1\ 0$
D. $4\ 3\ 2\ 1\ 0\ 5\ 6\ 7\ 8\ 9$

Your Answer: | **Correct Answer: B;C;D** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #10 Multiple Select Type | Award: 1 | Penalty: 0 | DS

Consider an array-based queue implementation. Suppose we store the following variable to implement a queue

1. front
2. back
3. SIZE of the queue (number of values **currently** in the queue)
4. CAPACITY of array (Maximum number of elements we can put in the queue)

Now, instead of storing all the above variables, we want to store fewer variables out of four variables.

Which of the following is/are correct options which tell sufficient variables to store to implement the queue?

A. front, back, and SIZE
B. front, back and CAPACITY
C. front and back
D. front and CAPACITY

Your Answer: | **Correct Answer: B** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #11 Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

Consider the following piece of code which finds the size of the queue when the queue is implemented using a circular array. Back and front are pointers pointing to the back and front of the queue respectively, and cap is the capacity of the array.

Fill the return statement in the else part so that code works correctly to find the size of the queue.

```
    if (back >= front)
        return back - front;
    else
        return _______
```

A. (back+cap)−front
B. cap − (back+front)
C. front−back
D. cap − (back−front)

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #12 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

Here is an **INCORRECT** pseudocode for the algorithm which is supposed to determine whether a sequence of parentheses is balanced:

```
    Declare a character stack
    while ( more input is available)
    {
        read a character
 5.     if ( the character is a '(' )
            push it on the stack
        else if ( the character is a ')' and the stack is not empty )
            pop a character off the stack
        else
10.         print "unbalanced" and exit
    }
    print "balanced"
```

Which of these unbalanced sequences does the above code think is balanced?

A. (((())
B. ())(()
C. ((()()))
D. ((()))()

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #13 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

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.

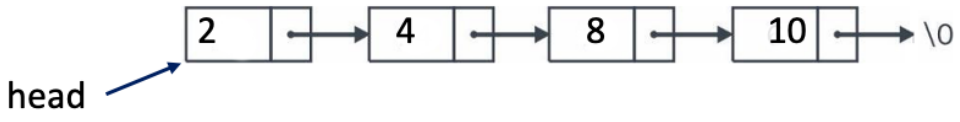
Which of the following implementations of the list should be used?

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

Your Answer: | Correct Answer: D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #14 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

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;
5.  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. $2 \rightarrow 6 \rightarrow 4 \rightarrow 8 \rightarrow 10$

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #15 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | DS

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;
5.      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

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #16 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

Suppose we have a Node data structure declared as follows:

```
    struct Node
    {
        int item;
        Node *next;
5.  };
```

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;
        }
 5.     else {
            if (arg -> item >0) return 1 + func(arg ->next);
            else
                return func(arg -> next);
        }
10. }
```

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

Your Answer: | **Correct Answer: B** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #17

Multiple Select Type | Award: 2 | Penalty: 0 | DS

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

        //The list is restored to its initial state

        insert2(&head,9);
25.     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 $1, 2, 3, 4, 5$

Your Answer: | **Correct Answer: B;C** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #18 | Multiple Select Type | Award: 2 | Penalty: 0 | DS

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) {
 5.             Node * newNext = current -> next -> next;
                while (newNext != NULL && newNext -> data == marker) {
                    newNext = newNext -> next;
                }
                current -> next = newNext;
10.         }
            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 : $6 \rightarrow 3$.

Your Answer: | Correct Answer: A;C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #19 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

Consider the following function recursive_remove().

```
    Node* recursive_remove(Node* head, int x)
    {
        if (head == NULL)
            return NULL;
 5.     if (head->value == x)
        {
            Node* tmp = head->next;
            free(head);
            return recursive_remove(tmp, x);
10.     }
        else
        {
            head->next = recursive_remove(head->next, x);
            return head;
15.     }
    }
```

If we call the function recursive_remove(head,2) 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

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #20 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

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

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

    int fun(Node *head){
        if(head== NULL) return 1;
        Node *p,*q;
10.     p = head;
        q = p->next;
        while(q!=NULL && q!=p){
            q = q->next;
        if(q==NULL) return 1;
15.     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

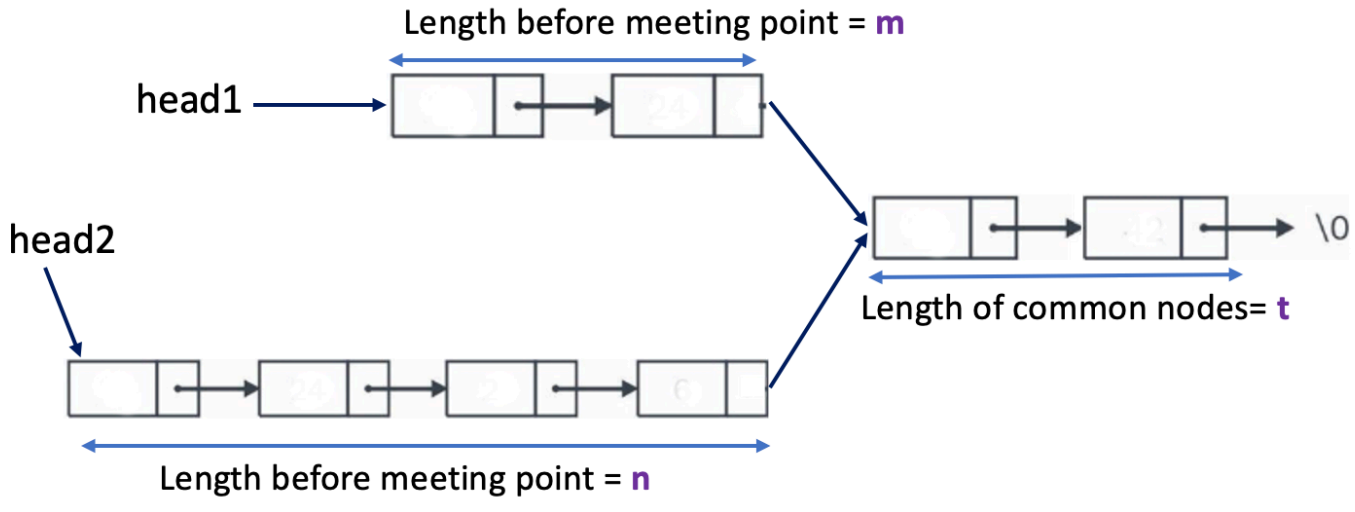
Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #21 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

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 $O(\mathrm{MN})$.

Your Answer: | **Correct Answer: C** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #22 Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

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

Your Answer: | **Correct Answer: C** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #23 Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

Suppose we have a min-heap holding n numbers. We want to visit every number in the heap that is less than a given value x. Let m is the number of elements that satisfy the condition. Let the height of the heap is h then what is the worst-case time complexity for such traversal?

A. $\theta(h)$
B. $\theta(m)$
C. $\theta(n)$
D. $\theta(h+m)$

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #24 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Algorithms

Rank the following functions by increasing order of growth; that is, find an arrangement g_1, g_2, g_3, g_4 of the functions satisfying $g_1 = O(g_2)$, $g_2 = O(g_3)$, $g_3 = O(g_4)$. (For example, the correct ordering of n^2, n^4, n, n^3 is n, n^2, n^3, n^4)

$$f_1 = (n!)^{1/n} \quad f_2 = \log n^n \quad f_3 = n^{n^{1/2}} \quad f_4 = n \log n \log\log n$$

A. $f1, f2, f3, f4$
B. $f3, f2, f4, f1$
C. $f1, f2, f4, f3$
D. $f2, f4, f1, f3$

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 02sec | Discuss

Q #25 | Multiple Select Type | Award: 2 | Penalty: 0 | Algorithms

Consider $\text{Iterated logarithm}$ of n, written $\log^* n$ (usually read "$\log$ star n"), is the number of times the logarithm (base 2) function must be iteratively applied before the result is less than or equal to 1. Mathematically,

$$\lg^* N = \min\{k \mid \underbrace{\lg \lg \cdots \lg}_{k} N \leq 1\}$$

For example-

$$\lg^* 2 = 1,$$
$$\lg^* 4 = 2,$$
$$\lg^* 16 = 3,$$
$$\lg^* 65536 = 4,$$
$$\lg^* \left(2^{65536}\right) = 5$$

Which of the following is/are correct about the implementation of $\log *n$?

In all options, $\log()$ is a function which calculates $\log$ base 2 of a given number.

A.

```
int iteratedLog(double n)
{
    if (n > 1.0) {
        return 1 + iteratedLog( log(n));
    }
    else return 0;
}
```

B.

```
    int iteratedLog(double n)
    {
        int count=0;
        while (n > 1.0) {
 5.         n = log(n);
            count++;
        }
        return count;
    }
10.
```

C. $\log * n = \begin{cases} 0 & \text{if} \quad n \leq 1 \\ 1 + \log * (\log n) & \text{if} \quad n > 1 \end{cases}$

D. $\log^* 2^m = 1 + \log^* m$ for $m > 1$

Your Answer: | Correct Answer: A;B;C;D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #26 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

Let head is a pointer to first node of following linked list -

$$4 \rightarrow 2 \rightarrow 5 \rightarrow 2$$

```
    int sumlinked(Node *head)
    {
        int sum=0;
        int i=0;
 5.     for (; head != NULL; head = head→ next) {
            if (i % 2 == 0)
                sum += head→ key;
            else
                sum -= head→ key;
10.             i++;
        }
        return sum;
    }
```

What will be the output if head is passed to above function?

A. 3
B. 5
C. -3
D. -5

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #27 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Algorithms

What will be the time complexity of $\text{fun}()$?

```
    int fun(int n) {
        int count = 0;
        for (int i = n; i > 0; i /= 2)
            for (int j = 0; j < i; j++)
5.              count += 1;
        return count;
    }
```

A. $\Theta(n)$
B. $\Theta(n \log n)$

C. $\Theta\left(n^2\right)$
D. $\Theta\left((\log n)^2\right)$

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #28 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Algorithms

What is the time complexity of the code given?

```
    for (i = n; i > 0; i--) {
        for (j = 1; j < n; j = j * 2) {
            for (k = 0; k < j; k++) {
                printf("hi");
5.          }
        }
    }
```

A. $\Theta(n)$
B. $\Theta\left(n^2\right)$
C. $\Theta\left(n^3\right)$
D. $\Theta\left(n^2 \log n\right)$

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #29 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | DS

Consider N employee records to be stored in memory for on-line retrieval. Each employee record is uniquely identified by a social security number. Consider the following ways to store the N records.

I. An array sorted by social security number
II. A linked list sorted by social security number
III. A linked list not sorted
IV. A balanced binary search tree with social security number as key

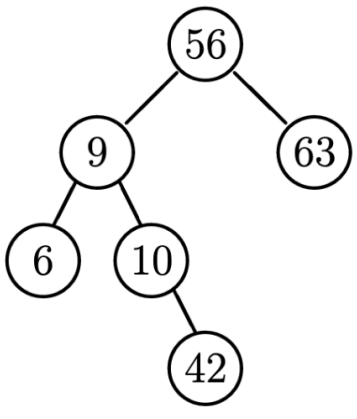
For the structures I-IV, respectively, the average time for an efficient program to find an employee record, given the social security number as key, is which of the following?

	I	II	III	IV
A.	$\mathrm{O}(\log \mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\log \mathrm{N})$
B.	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\log \mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\mathrm{N})$
C.	$\mathrm{O}(\log \mathrm{N})$	$\mathrm{O}(\log \mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(1)$
D.	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(\mathrm{N})$	$\mathrm{O}(1)$

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #30 | Numerical Type | Award: 2 | Penalty: 0 | DS

Say the following tree was obtained by inserting the element 42 into an AVL tree. The tree no longer satisfies the AVL invariant, but the invariant can be re-established by performing two rotate operations.



What will be the root of tree after rotations?

Your Answer: | **Correct Answer: 10** | Not Attempted | Time taken: 00min 00sec | Discuss

Copyright & Stuff