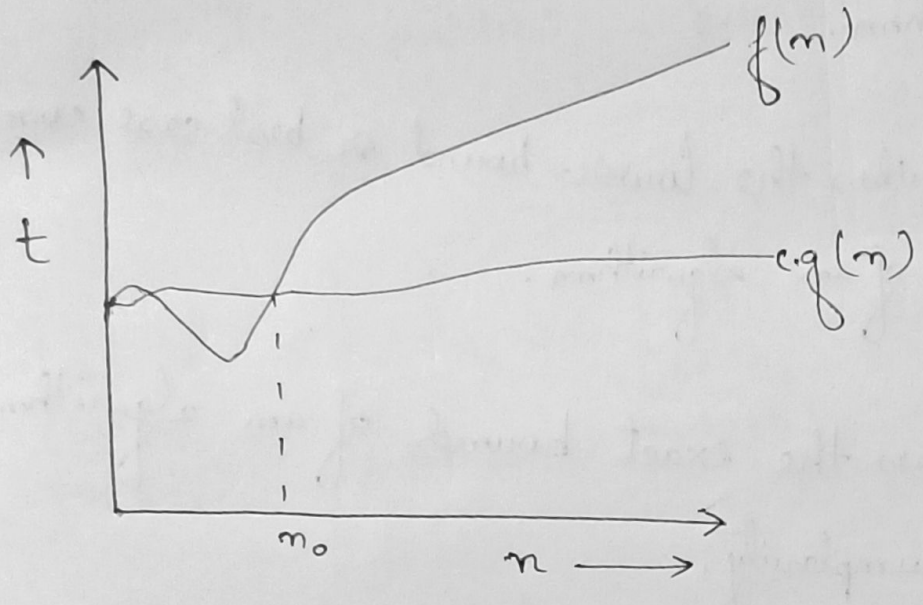
Suggestions

1) Omega notation is used in algorithm analysis to describe the lower bound of a algorithm's running time. Mathematically, $f(n) \geq c \cdot g(n)$.



Eg.:-

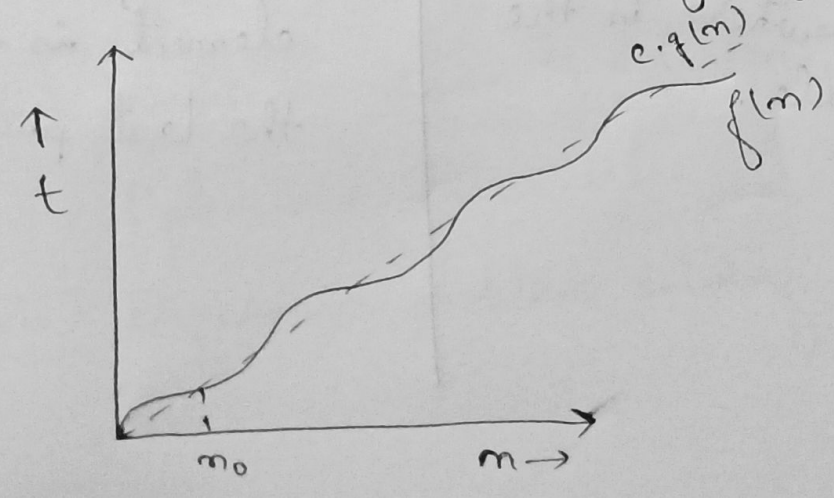
$f(n) = 3n + 2$

$g(n) = n$

Let, $c = 3$, $n_0 = 1$

$\therefore 3n + 2 \geq 3n$ for all $n \geq 0$

2) In algorithm analysis, worst case scenario refers to the situation where the algorithm takes the longest time to complete. Mathematically, $f(n) \leq c \cdot g(n)$



3) Asymptotic notations are the notations used to describe the behaviour of time or space complexity. Common asymptotic notations are :-

(i) Big oh :- Describes the upper-bound or worst case complexity of an algorithm.

(ii) Omega :- Describes the lower-bound or best case complexity of an algorithm.

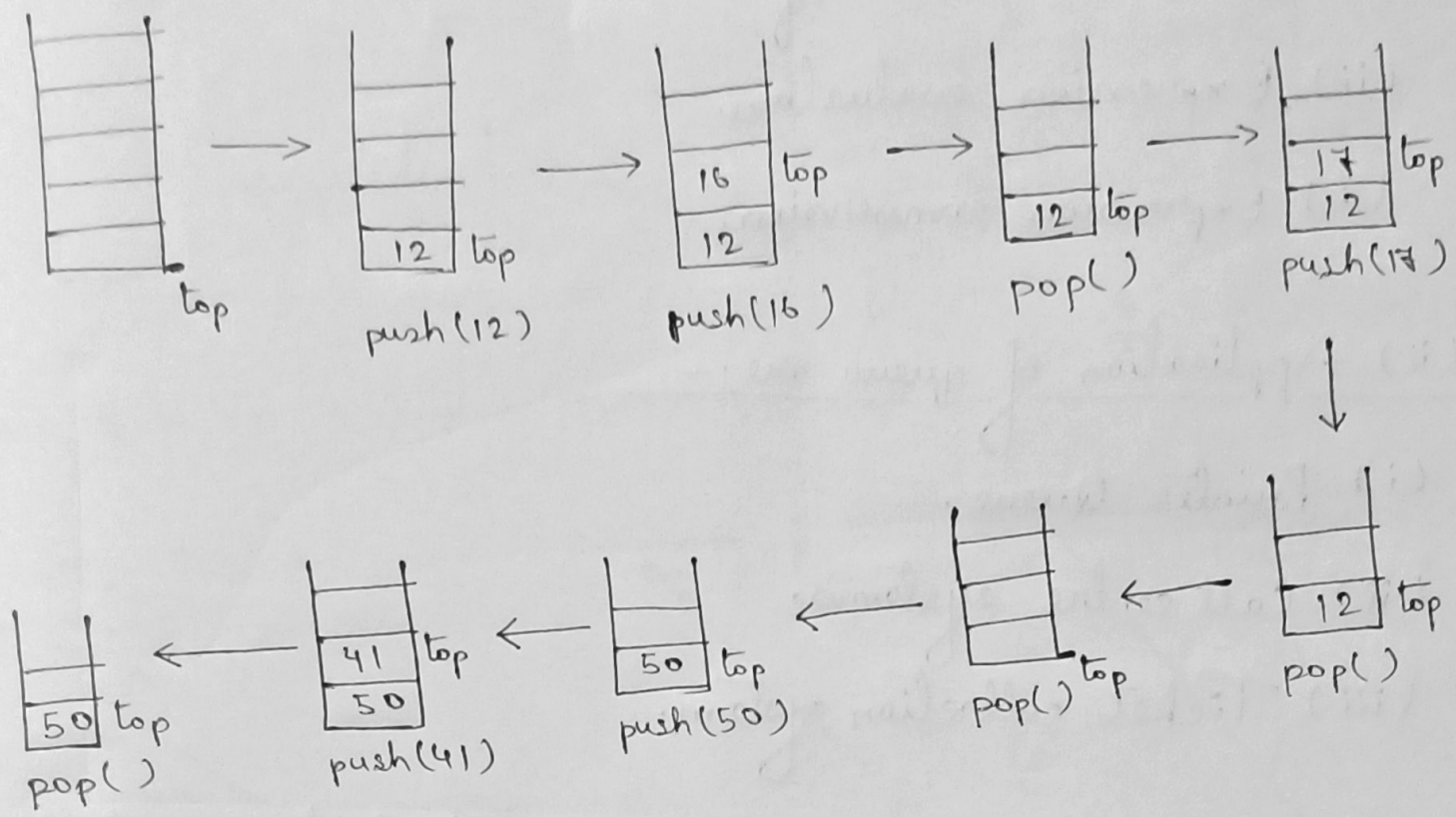
(iii) Theta :- Describes the exact bound of an algorithm's time complexity.

4)

Best Case	Average Case	Worst Case
(i) The algorithm performs fastest.	(i) The algorithm performs as expected.	(i) The algorithm perform slowest.
(ii) Minimum time taken.	(ii) Expected time taken.	(ii) Maximum time taken.
(iii) Describes lower bound.	(iii) Describes expected time.	(iii) Describes upper bound.
(iv) Target element is at first position.	(iv) Target element is somewhere in the middle.	(iv) Target element is in the last position.

5) Theta notation is used for average case.

6)



7) (i) Time Complexity :- Time complexity is the measure of the amount of time an algorithm takes to complete as as function of the input size.

(ii) Space Complexity :- Space complexity is the ~~most~~ measure of the amount of memory an algorithm requires as a function of the input size.

8) The advantages of non-linear data structure in computation are :-

(i) Efficient search operations.

(ii) Faster insertion & deletion.

(iii) Optimised traversal.

(iv) Dynamic problem solving.

9) (i) Application of stack are :-

(i) Parenthesis matching.

(ii) Expression evaluation.

(iii) Expression conversion.

(ii) Application of queue are :-

(i) Printer Queue.

(ii) Call centre systems.

(iii) Ticket collection system.

10) (i) Simple Queue :- Elements are added from rear and removed from front.

(ii) Circular Queue :- Last position is connected back to the first position.

(iii) Priority Queue :- Each element is associated with a priority.

(iv) Double-Ended Queue :- Insertion & deletion can be performed in both ways.

11) (i) Checks if stack is full.

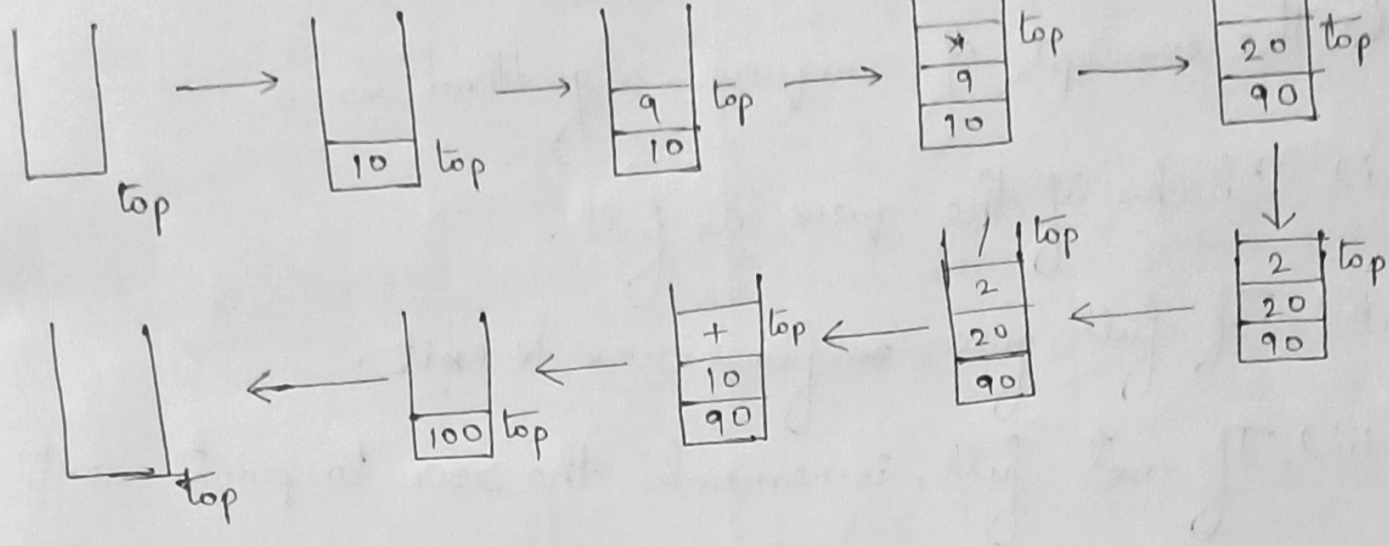
(ii) If full, gives error & exit.

(iii) If not full, increements top to point next empty space.

(iv) Adds data element.

(v) Returns success.

12)



∴ 10 9 * 20 2 / + = 100

13)

Input

A

+

B

*

(

Stack

+

+

*

Postfix

A

A

AB

AB+

13) Input

Input	Stack	Postfix
(	(	
A	(	A
+	(+	A
B	(+	AB
)		AB+
*	*	AB+
(	*(	AB+
C	*(	AB+C
-	*(-	AB+C
D	*(-	AB+CD
)	*	AB+CD-
	empty	AB+CD-*

15) The concept of enqueue algorithm is :-

(i) Checks if the queue is full.

(ii) If full, gives ~~overflow~~ error & exit.

(iii) If not full, increments the rear to point next empty space.

(iv) Adds data element.

(v) Returns Success.

16) The concept of dequeue algorithm is :-

(i) Checks if the queue is empty.

(ii) If ~~full~~ empty, gives error & exit.

(iii) If not empty, retreive the front element.

(iv) Increments front to the next position.

(v) Returns success.

17) The condition to check whether a circular queue is empty or full :-

(i) Empty :- front == -1.

(ii) Full :- (rear + 1) % size == front

18) The key advantages of using a linked list to implement a queue over an array is :-

(i) Dynamic Size :- Linked list does not have specific size but an array has a specific size.

(ii) No worry of overflow :- Linked list grows dynamically so there are no worry of overflow whereas array does not grow dynamically.

19) Operations of stack are :-

(i) isEmpty() :- Checks if the stack is empty or not.

(ii) isFull() :- " " " " " full.

(iii) push() :- Inserts an element in the stack.

(iv) pop() :- Deletes " " from " " .

(v) peek() :- Gives the topmost element on the stack.

Operations on queue are :-

(i) isEmpty() :- Checks if the queue is empty.

(ii) isFull() :- " " " " full.

(iii) enqueue :- Adds element to the rear of the queue.

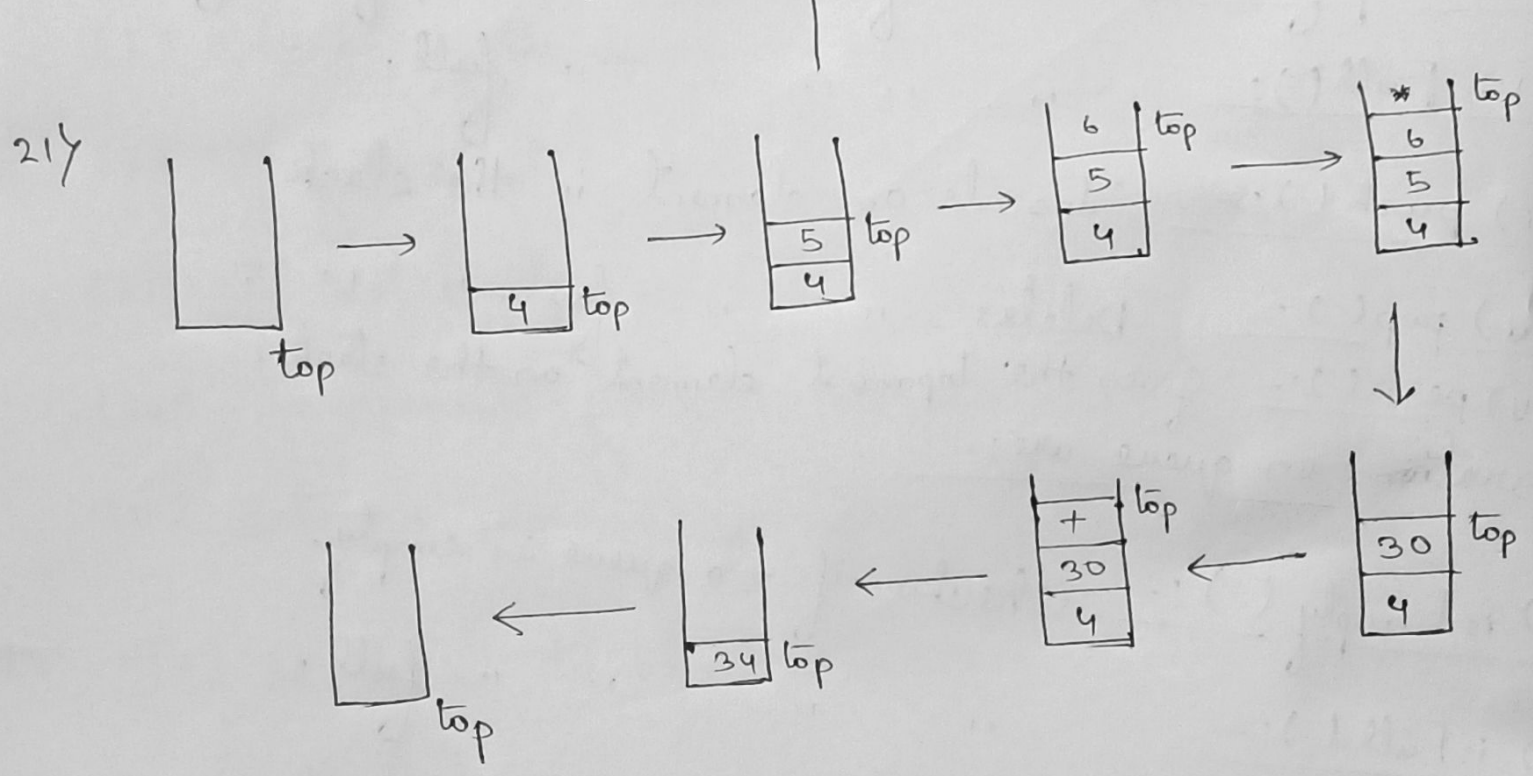
(iv) dequeue :- Removes element from the front of the queue.

(v) peek() :- Returns the front element in the queue.

20)

Linear DS	Non-Linear DS
(i) Elements are arranged sequentially.	(i) Elements are arranged non-sequentially.
(ii) Contiguous memory allocation.	(ii) Non-contiguous memory allocation.
(iii) Linear traversal.	(iii) Complex traversal.
(iv) Simple applications.	(iv) Complex applications.
(v) Follows a specific order.	(v) No strict order.

21)



∴ 4 5 6 * + = 34

22) The advantages of using postfix notation over infix notation are :-

(i) No need for parenthesis.

(ii) Easier & Faster to evaluate.

(iii) No need to handle operator precedence.

(iv) Easier for compilers.

23) (i) The concept of enqueue in circular queue is:-

(i) Checks for full.

(ii) If full, gives error & exit.

(iii) If not full, add element to the rear.

(iv) Update the rear pointer by modulo operator.

(v) Returns success.

(ii) The concept of dequeue in circular queue is:-

(i) Check for 'empty'.

(ii) If empty, gives error & exit.

(iii) If not empty, remove element from the front.

(iv) Update front pointer using modulo operator.

(v) Returns success.

24) For Big oh explanation go to the answer of question no. 2.

$f(n) = 3n^2 + 3n + 5$

$g(n) = n^2$

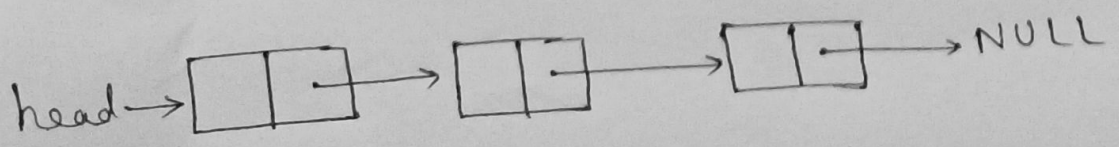
Let, $c = 10$, $n_0 = 2$

$\therefore 3n^2 + 3n + 5 \leq 10n^2$ for all $n \geq 2$

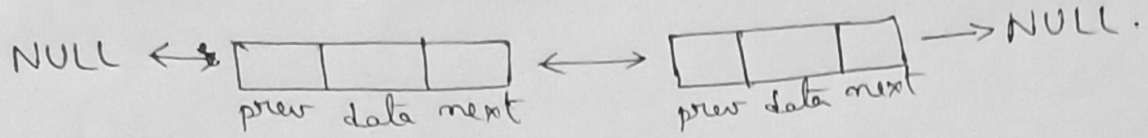
25)
```
if( pos > 1 && pos < nodectr ) {
        temp = prev = start;
        ctr = 1;
        while( ctr < pos ) {
            prev = temp;
            temp = temp -> next;
            ctr++;
        }
    prev -> next = temp -> next;
    free( temp );
}
```

26)

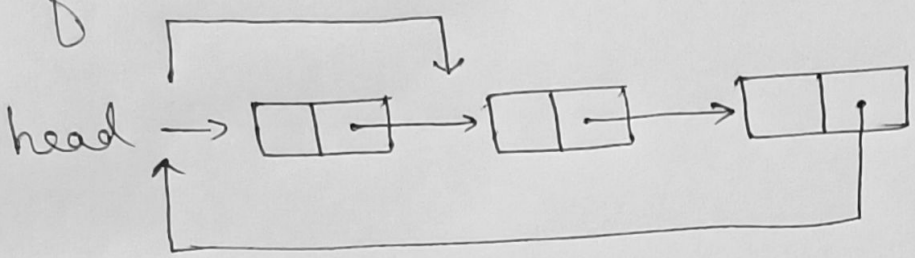
27) (i) Singly linked list:- A singly linked list consists of nodes where each node consists data and pointer to next node. It ends with null reference.



(ii) Doubly linked list :- A doubly linked list has nodes that contain a reference to both previous & next node.



(iii) Circular linked list :- It's last node points back to first node.



28) & Stacks

(i) Last in First Out.

(ii) Elements are removed from top.

(iii) Elements are added to top.

(iv) push() & pop() operations.

Queues

(i) First in First out.

(ii) Element are removed from front.

(iii) Elements are added to rear.

(iv) enqueue() & dequeue() operations.

29) FIFO is a principle used in queues where the first element added to the queue should be removed first.

Eg.:- enqueue(1) = [1]
enqueue(2) = [1, 2]
dequeue() = [2] (1 removed first)