(Q) Stack as ADT

→ Stack is a LIFO (Last In First out) structure. It is an ordered list of same type of elements. A stack is a linear list where all insertions and deletions are permitted at only one end of the list, namely the front end of the list called the top. The operations on the stack are :-

1. Initialize() - Make a stack empty
2. Empty() - To determine if stack is empty or not.
3. Full() - To determine if stack is full or not.
4. Push() - If a stack is not full then push a new element at the top of the stack
5. Pop() - If a stack is not empty, then pop the element from top of the stack
6. Peek() - Returns top element of stack.

A pop() on an empty stack is considered to be an error. A push operation on a stack that is full is considered to be an error.

Structure used for stack :-

```
typedef struct stack {
    int data[SIZE]; //it can be of any data type.
    int top;
} stack;
```

(Q) Applications of Stack

→ 1. Expression Conversion
   (a) Infix to Postfix (b) Infix to Prefix
   (c) Postfix to infix (c) Prefix to infix
2. Expression evaluation
3. Parsing
4. Simulation of recursion
5. Function call.

(Q) Explain infix, postfix, prefix expression with an example.

→ There are three popular methods for representation of an expression.

(a) Infix	x+y	operator between operands
(b) Prefix	+xy	operator before operands
(c) Postfix	xy+	operator after operands

Example:

Infix	x+y*z
Prefix	+x*yz
Postfix	xyz*+

(a) Evaluation of Infix Expression

⇒ Infix expressions are evaluated left to right but operator precedence must be taken into account. To evaluate x+y*z, y and z will be multiplied first and then added to x.

Infix are not used for representation of an expression inside computer due to complexity of handling precedence

(b) Evaluation of Prefix Expression.

⇒ Consider an example for evaluation of prefix exp.

+5 * 3 2

Find an operator from right to left and perform the operation

+5 *32

First operator is *, therefore 3*2 is multiplied and expression becomes +56.

First operator is +, therefore 5+6 is added and expression becomes 11.

(c) Evaluation of postfix expression

⇒ 5 3 2 * +

Find first operator from left to right and perform operation.

First operator is *, therefore 3*2 is multiplied and expression becomes 56 +

First operator is +, therefore 5+6 is added and expression becomes 11.

(Q) Compare stacks and queue.

Stack	Queue
① Stack is last in first out (LIFO) i.e which is entered will be retreived first	① Queue is first in first out i.e which is entered first will be retreived first (FIFO)
② Stack is linear data structure which follows LIFO	② Queue is linear data structure which follows FIFO
③ Insertion & deletion are possible through one end called top	③ Insertion are at rear end and deletions are from front end in a queue.
④ Eg:- Books in library	Eg:- Cinema ticket counter

(Q.) Use of stack in function call

→ Nested calls of functions are managed through stack. When function is invoked, memory is allocated on stack for local variables and return address. On termination of execution of function, memory is deallocated. Details regarding called functions are stored in a specific structure known as activation record. It consists of:

1. Storage space for local variables
2. Definition of function
3. Return address
4. A pointer to activation record.

Activation records are stored on stack. When function is called, its activation record is pushed into stack and control is transferred to called program. When function execution completes, control returns to caller function. It obtains return address from return address field of activation record.

(Q) Queue as ADT

→ Queue is a special kind of a list where items are inserted at one end and deleted from the other end. Queue is a FIFO (First In First Out) list.

Operations on Queue includes:

1. initialize() - Initializes a queue by setting value of rear and front to -1
2. enqueue(): Inserts element at rear end of queue.
3. dequeue(): Deletes front element and returns same.

④ empty() : returns true if queue is empty
⑤ full() : returns true if queue is full.
⑥ print() : prints queue elements.

Array representation of queue requires 3 entities:
(a) Array to hold elements
(b) Variable to hold index of front element
(c) Variable to hold index of rear element.

```
#define MAX 30
typedef struct queue {
    int data[MAX];
    int front, rear;
} queue;
```

(Q) Advantages of using circular queue

→ One potential problem with implementation of queue using simple array is that the queue may appear to be full although there might be some space in the queue. The simple solution to that whenever rear gets to end of the array, it is wrapped around to the start. Now the array can be thought of as a circle - the first position follows last position.

(Q) Basic Tree Terminology.

→ A tree is a collection of elements called nodes, one of which is distinguished as root, along with relation 'parent'. The root of each subtree is said to be child of r and r is the parent of each subtree root.

① Root :- It is a special node in tree and entire tree is refrenced through it.

② Parent :- Immediate predecessor of a node.

③ Child :- Immediate successors of a node

④ Siblings : Nodes with same parents are called siblings

⑤ Path : Number of edges from source node to destination node.

⑥ Degree of node : Degree of a node is the number of children of a node.