Core - CSE

ANS-1- OOPS:

Object-oriented programming combines a group of variables (properties) and functions (methods) into a unit called an "object." These objects are organized into classes where individual objects can be grouped together. OOP can help you consider objects in a program's code and the different actions that could happen in relation to the objects.

This programming style widely exists in commonly used programming languages like Java, C++ and PHP. These languages help simplify the structure and organization of software programs. Programmers often use OOP when they need to create complex programs.

>> Principles of object-oriented programming:

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

>> Encapsulation:

- The different objects inside of each program will try to communicate with each other automatically. If a programmer wants to stop objects from interacting with each other, they need to be encapsulated in individual classes. Through the process of encapsulation, classes cannot change or interact with the specific variables and functions of an object.

- Just like a pill "encapsulates" or contains the medication inside of its coating, the principle of encapsulation works in a digital way to form a protective barrier around the information that separates it from the rest of the code. Programmers can replicate this object throughout different parts of the program or other programs.

>> Abstraction:

- Abstraction is like an extension of encapsulation because it hides certain properties and methods from the outside code to make the interface of the objects simpler. Programmers use abstraction for several beneficial reasons. Overall, abstraction helps isolate the impact of changes made to the code so that if something goes wrong, the change will only affect the variables shown and not the outside code.

>> Inheritance:

- Using this concept, programmers can extend the functionality of the code's existing classes to eliminate repetitive code. For instance, elements of HTML code that include a text box, select field and checkbox have certain properties in common with specific methods.

- Instead of redefining the properties and methods for every type of HTML element, you can define them once in a generic object. Naming that object something like "HTML Element" will cause other objects to inherit its properties and methods so you can reduce unnecessary code.

- The main object is the superclass and all objects that follow it are subclasses. Subclasses can have separate elements while adding what they need from the superclass.

>> Polymorphism:

- This technique meaning "many forms or shapes" allows programmers to render multiple HTML elements depending on the type of object. This concept allows programmers to redefine the way something works by changing how it is done or by changing the parts in which it is done. Terms of polymorphism are called overriding and overloading.

ANS-2- TCP/IP MODEL:

The **TCP/IP model** is a concise version of the OSI model. It contains four layers, unlike seven layers in the OSI model. The layers are:

- Network Access/Link Layer
- Internet Layer
- Host-to-Host/Transport Layer
- Process/Application Layer

>> Network Access Layer/ Link Layer:

This layer corresponds to the combination of Data Link Layer and Physical Layer of the OSI model. It looks out for hardware addressing and the protocols present in this layer allows for the physical transmission of data.
We just talked about ARP being a protocol of Internet layer, but there is a conflict about declaring it as a protocol of Internet Layer or Network access layer.

The first layer is the Process layer on the behalf of the sender and Network Access layer on the behalf of the receiver. It is described as residing in layer 3, being encapsulated by layer 2 protocols.

>> Internet Layer:

This layer parallels the functions of OSI's Network layer. It defines the protocols which are responsible for logical transmission of data over the entire network. The main protocols residing at this layer are:

- IP – stands for Internet Protocol and it is responsible for delivering packets from the source host to the destination host by looking at the IP addresses in the packet headers. IP has 2 versions: IPv4 and IPv6.
IPv4 is the one that most of the websites are using currently. But IPv6 is growing as the number of IPv4 addresses are limited in number when compared to the number of users.

- ICMP – stands for Internet Control Message Protocol. It is encapsulated within IP datagrams and is responsible for providing hosts with information about network problems.

- ARP – stands for Address Resolution Protocol. Its job is to find the hardware address of a host from a known IP address. ARP has several types: Reverse ARP, Proxy ARP, Gratuitous ARP and Inverse ARP.

>> Host-to-Host Layer/ Transport Layer:

This layer is analogous to the transport layer of the OSI model. It is responsible for end-to-end communication and error-free delivery of data. It shields the upper-layer applications from the complexities of data. The two main protocols present in this layer are:

- Transmission Control Protocol (TCP) – It is known to provide reliable and error-free communication between end systems. It performs sequencing and segmentation of data. It also has acknowledgment feature and controls the flow of the data through flow control mechanism. It is a very effective protocol but has a lot of overhead due to such features. Increased overhead leads to increased cost.

- User Datagram Protocol (UDP) – On the other hand does not provide any such features. It is the go-to protocol if your application does not require reliable transport as it is very cost-effective. Unlike TCP, which is connection-oriented protocol, UDP is connectionless.

>> Application Layer/ Process Layer:

This layer performs the functions of top three layers of the OSI model: Application, Presentation and Session Layer. It is responsible for node-to-node communication and controls user-interface specifications. Some of the protocols present in this layer are: HTTP, HTTPS, FTP, TFTP, Telnet, SSH, SMTP, SNMP, NTP, DNS, DHCP, NFS, X Window, LPD. Have a look at Protocols in Application Layer for some information about these protocols. Protocols other than those present in the linked article are:

- HTTP and HTTPS – HTTP stands for Hypertext transfer protocol. It is used by the World Wide Web to manage communications between web browsers and servers. HTTPS stands for HTTP-Secure. It is a combination of HTTP with SSL (Secure Socket Layer). It is efficient in cases where the browser needs to fill out forms, sign in, authenticate and carry out bank transactions.

- SSH – SSH stands for Secure Shell. It is a terminal emulations software similar to Telnet. The reason SSH is more preferred is because of its ability to maintain the encrypted connection. It sets up a secure session over a TCP/IP connection.

- NTP – NTP stands for Network Time Protocol. It is used to synchronize the clocks on our computer to one standard time source. It is very useful in situations like bank transactions. Assume the following situation without the presence of NTP. Suppose you carry out a transaction, where your computer reads the time at 2:30 PM while the server records it at 2:28 PM. The server can crash very badly if it's out of sync.

ANS-3- BINARY TREE:

>> Input:

- In order => {N, M, P, O, Q}

- Post order => {N, P, Q, O, M}

>> Code(with all the comments explaining steps in bold):

```
# Class to store a binary tree node
class Node:
    def _init_(self, key):
        self.key = key


# Recursive function to perform inorder traversal on a given binary tree
def inorderTraversal(root):

    if root is None:
        return

    inorderTraversal(root.left)
    print(root.key, end=' ')
    inorderTraversal(root.right)

# Recursive function to perform postorder traversal on a given binary tree
def postorderTraversal(root):
```

```
    if root is None:
        return

    postorderTraversal(root.left)
    postorderTraversal(root.right)
    print(root.key, end=' ')


# Recursive function to construct a binary tree from a given
# inorder and postorder traversals
def construct(start, end, postorder, pIndex, dict):

    # base case
    if start > end:
        return None, pIndex

    # Consider the next item from the end of a given postorder sequence.
    # This value would be the root node of a subtree formed by sequence
    # `inorder[start, end]`.
    root = Node(postorder[pIndex])
    pIndex = pIndex - 1

    # search the current node index in inorder sequence to determine
    # the boundary of the left and right subtree of the current node
    index = dict[root.key]

    # recursively construct the right subtree
    root.right, pIndex = construct(index + 1, end, postorder, pIndex, dict)

    # recursively construct the left subtree
    root.left, pIndex = construct(start, index - 1, postorder, pIndex, dict)

    # return the root node
    return root, pIndex


# Construct a binary tree from inorder and postorder traversals.
# This function assumes that the input is valid, i.e., given
# inorder and postorder sequences forming a binary tree.
def constructTree(inorder, postorder):

    # get size
    n = len(inorder)

    # dictionary is used to efficiently find the index of any element in
    # a given inorder sequence
    dict = {}
    for i, e in enumerate(inorder):
        dict[e] = i

    # `pIndex` stores the index of the next unprocessed node from the end
    # of the postorder sequence
    pIndex = n - 1
    return construct(0, n - 1, postorder, pIndex, dict)[0]
```

```
if _name_ == '_main_':

    ''' Construct the following tree
                    M
                  /     \
                N         O
                        /     \
                      P         Q
       '''

    inorder = [N, M, P, O, Q]
    postorder = [N, P, Q, O, M]

    root = constructTree(inorder, postorder)

    # traverse the constructed tree
    print("Inorder traversal is ", end='')
    inorderTraversal(root)

    print("\nPostorder traversal is ", end='')
    postorderTraversal(root)
```

ANS-5- LRU Cache:

As a user you want pages to load as fast as possible. When a user opens a website, it opens the corresponding file on disk, read it in the HTML, and send it back over the network. This is how it works but it is pretty slow, since accessing disk takes a while.

So, if a lot of users request the same website, it's better to only read it from the disk once while keeping the page in memory so you can quickly send it out again when it is requested and that's simply all what a cache means and used for.

A cache is just a fast storage which means reading data from a cache takes less time than reading it from something else (like a hard disk) but caches are generally very small when compared to hard disks and far more costly as well. So, you can't fit everything in a cache and that's why you're still going to have to use larger and slower storage from time to time.

>> **We use two data structures to implement an LRU Cache.**

- **Queue:** which is implemented using a doubly linked list.

- **Hash:** with page number as key and address of the corresponding queue node as value.

>> Strengths:

- **Super fast accesses:** LRU caches store items in order from most-recently used to least-recently used. That means both can be accessed in O (1)O(1) time.

- **Super fast updates:** Each time an item is accessed, updating the cache takes O (1)O(1) time.

>> Weaknesses:

- **Space heavy:** An LRU cache storing n items requires a linked list of length n, and a hash map storing n items. That's O(n)O(n) space which is two data structures and space heavy.

ANS-6- Virtual Memory:

Virtual memory is a feature of an operating system that enables a computer to be able to compensate shortages of physical memory by transferring pages of data from random access memory to disk storage. This process is done temporarily and is designed to work as a combination of RAM and space on the hard disk.

>> **Example of Virtual Memory:**

VMM has to work to arise the new files on the hard disk, which are need (100 MB) such as (200 MB-100 MB) = 100 MB. VMM has responsible to deal in real memory, which is only 100 MB.

>> Advantages:

- The primary benefits of virtual memory include freeing applications from having to manage a shared memory space, ability to share memory used by libraries between processes, increased security due to memory isolation, and being able to conceptually use more memory than might be physically available, using the technique of paging or segmentation.

>> Disadvantages:

- Using virtual memory makes a computer run slower, as the processor has to wait while data is swapped between hard disk and RAM. As secondary storage devices have slower access times than RAM, the computer's processing performance can be severely impaired.

ANS-7- Deadlock Characteristics:

A process in operating systems requests different resources and uses them in the following way:

- Requests a resource
- Use the resource
- Releases the resource

A deadlock happens in an operating system when two or more processes need some resources to complete their execution that is held by another process.

>> There are four conditions that must be present for a deadlock to occur:

- Mutual Exclusion: There should be a resource that can only be held by one process at a time. So, the other processes which need the same resource can't have it.

- No Pre-emption: When a process can hold multiple resources and still request more resources from other processes which are holding them.

- Hold and Wait: When a resource cannot be pre-empted from a process by force. A process can only release a resource voluntarily when it is ready or finished with its execution.

- Circular Wait: When a process is waiting for the resource held by the second process, which is waiting for the resource held by the third process and so on, till the last process is waiting for a resource held by the first process making a circular chain of dependent processes.

ANS-8- NAT:

- NAT stands for **Network Address Translation**. Basically, NAT allows a single device like a router to act as a central source for an IP address to a public network and uses internal private IP addresses for the devices connected to the central router which means that only a single unique IP address is required to represent an entire group of devices to the rest of the network.

A router which is connected with the local network will be having an inside local IP addresses for the devices connected internally or privately to it and the router itself which is connected to the global network will be having a global IP address.

ARP:

- ARP stands for **Address Resolution Protocol**. It is a network protocol used to find out the hardware address i.e., MAC address of a device from an IP address. The sending device uses ARP to map IP addresses to MAC addresses. The device sends an ARP request containing the IP address of the receiving device.

ARP is one of the most important protocols of the network layer in the OSI model which helps in finding the MAC address (Media Access Control Address) from the IP address of the system i.e., the main duty of the ARP is to convert the 32-bit IP address (for IPv4) to 48-bit address i.e., the MAC address.