**Month-04 Test**

**Q1) List out different OOPS principles and explain?**

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 Python, Java, C++. These languages help simplify the structure and organization of software programs. Programmers often use OOP when they need to create complex programs. And must and should uses in every Projects

Object-oriented programming has four basic concepts Principles That is 1)encapsulation, 2)abstraction, 3)inheritance and 4)polymorphism.

* Encapsulation
* Abstraction
* Inheritance
* Polymorphism

1.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.

2.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.

3.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.

4.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.

**Q2. List out Layers of TCP/IP Model and explain?**

TCP/IP Model helps you to determine how a specific computer should be connected to the internet and how data should be transmitted between them. It helps you to create a virtual network when multiple computer networks are connected together. The purpose of TCP/IP model is to allow communication over large distances.

TCP/IP stands for Transmission Control Protocol/ Internet Protocol. TCP/IP Protocol Stack is specifically designed as a model to offer highly reliable and end-to-end byte stream over an unreliable internetwork.



**Layers of TCP/IP Model**

**1) Application Layer** - Application layer interacts with an application program, which is the highest level of OSI model. The application layer is the OSI layer, which is closest to the end-user. It means the OSI application layer allows users to interact with other software application.

Application-layer helps you to identify communication partners, determining resource availability, and synchronizing communication.

It allows users to log on to a remote host

This layer provides various e-mail services

This application offers distributed database sources and access for global information about various objects and services.

**2) Transport Layer** - Transport layer builds on the network layer in order to provide data transport from a process on a source system machine to a process on a destination system. It is hosted using single or multiple networks, and also maintains the quality-of-service functions.

It determines how much data should be sent where and at what rate. This layer builds on the message which are received from the application layer. It helps ensure that data units are delivered error-free and in sequence.

Transport layer helps you to control the reliability of a link through flow control, error control, and segmentation or de-segmentation.

The transport layer also offers an acknowledgment of the successful data transmission and sends the next data in case no errors occurred. TCP is the best-known example of the transport layer.

It divides the message received from the session layer into segments and numbers them to make a sequence.

Transport layer makes sure that the message is delivered to the correct process on the destination machine.

It also makes sure that the entire message arrives without any error else it should be retransmitted.

**3) Internet Layer** - An internet layer is a second layer of TCP/IP layes of the TCP/IP model. It is also known as a network layer. The main work of this layer is to send the packets from any network, and any computer still they reach the destination irrespective of the route they take.

The Internet layer offers the functional and procedural method for transferring variable length data sequences from one node to another with the help of various networks.

Message delivery at the network layer does not give any guaranteed to be reliable network layer protocol.

Routing protocols

Multicast group management

Network-layer address assignment.

**4) The Network Interface Layer** - Network Interface Layer is this layer of the four-layer TCP/IP model. This layer is also called a network access layer. It helps you to defines details of how data should be sent using the network.

It also includes how bits should optically be signaled by hardware devices which directly interfaces with a network medium, like coaxial, optical, coaxial, fiber, or twisted-pair cables.

A network layer is a combination of the data line and defined in the article of OSI reference model. This layer defines how the data should be sent physically through the network. This layer is responsible for the transmission of the data between two devices on the same network.

**Q3. Construct a binary tree by using postorder and inorder sequences given below.**

**Inorder: N, M, P, O, Q**

**Postorder: N, P, Q, O,**

Given Postorder and Inorder traversals, construct the tree

**Examples:**

Input:

in[] = {2, 1, 3}

post[] = {2, 3, 1}

Output: Root of below tree

1

/ \

2 3

Input:

in[] = {4, 8, 2, 5, 1, 6, 3, 7}

post[] = {8, 4, 5, 2, 6, 7, 3, 1}

Output: Root of below tree

1

/ \

2 3

/ \ / \

4 5 6 7

\

8

**Another example:**

# Python3 program to construct tree using

# inorder and postorder traversals

# Helper function that allocates

# a new node

class newNode:

    def \_\_init\_\_(self, data):

        self.data = data

        self.left = self.right = None

# Recursive function to construct binary

# of size n from Inorder traversal in[]

# and Postorder traversal post[]. Initial

# values of inStrt and inEnd should be

# 0 and n -1. The function doesn't do any

# error checking for cases where inorder

# and postorder do not form a tree

def buildUtil(In, post, inStrt, inEnd, pIndex):

    # Base case

    if (inStrt > inEnd):

        return None

    # Pick current node from Postorder traversal

    # using postIndex and decrement postIndex

    node = newNode(post[pIndex[0]])

    pIndex[0] -= 1

    # If this node has no children

    # then return

    if (inStrt == inEnd):

        return node

    # Else find the index of this node

    # in Inorder traversal

    iIndex = search(In, inStrt, inEnd, node.data)

    # Using index in Inorder traversal,

    # construct left and right subtress

    node.right = buildUtil(In, post, iIndex + 1,

                                  inEnd, pIndex)

    node.left = buildUtil(In, post, inStrt,

                        iIndex - 1, pIndex)

    return node

# This function mainly initializes index

# of root and calls buildUtil()

def buildTree(In, post, n):

    pIndex = [n - 1]

    return buildUtil(In, post, 0, n - 1, pIndex)

# Function to find index of value in

# arr[start...end]. The function assumes

# that value is postsent in in[]

def search(arr, strt, end, value):

    i = 0

    for i in range(strt, end + 1):

        if (arr[i] == value):

            break

    return i

# This function is here just to test

def preOrder(node):

    if node == None:

        return

    print(node.data,end=" ")

    preOrder(node.left)

    preOrder(node.right)

# Driver code

if \_\_name\_\_ == '\_\_main\_\_':

    In = [4, 8, 2, 5, 1, 6, 3, 7]

    post = [8, 4, 5, 2, 6, 7, 3, 1]

    n = len(In)

    root = buildTree(In, post, n)

   print("Preorder of the constructed tree :")

    preOrder(root)

# This code is contributed by PranchalK

**Output**

Preorder of the constructed tree :

1 2 4 8 5 3 6 7

**Q5. Explain LRU cache and its implementation by taking some examples and explaining all steps**

**A cache is just fast storage.** Reading data from a cache takes less time than reading it from something else (like a hard disk)

A Least Recently Used (LRU) Cache organizes items in order of use, allowing you to quickly identify which item hasn't been used for the longest amount of time.

Picture a clothes rack, where clothes are always hung up on one side. To find the least-recently used item, look at the item on the other end of the rack.

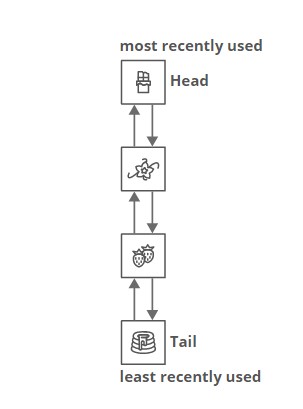
When a user requests a recipe, you open the corresponding file on disk, read in the HTML, and send it back over the network. This works, but it's pretty slow, since accessing disk takes a while.

Ideally, if lots of users request the same recipe, you'd like to only read it in from disk once, keeping the page in memory so you can quickly send it out again when it's requested. Bam. You just added a cache.

**LRU Cache Implementation**

An LRU cache is built by combining two data structures: [a doubly linked list](https://www.interviewcake.com/concept/linked-list) and [a hash map](https://www.interviewcake.com/concept/hash-map).

We'll set up our linked list with the most-recently used item at the head of the list and the least-recently used item at the tail:

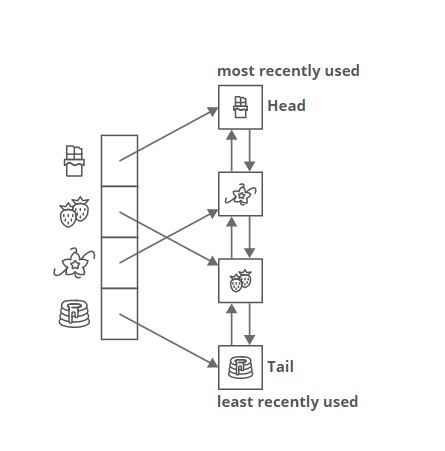


This lets us access the LRU element in O(1)*O*(1) time by looking at the tail of the list.

What about accessing a specific item in the cache (for example, the chocolate cake recipe)?

In general, finding an item in a linked list is O(n)*O*(*n*) time, since we need to walk the whole list. But the whole point of a cache is to get quick lookups. How could we speed that up?

We'll add in a hash map that maps items to linked list nodes:



That lets us find an element in our cache's linked list in O(1)*O*(1) time, instead of O(n)*O*(*n*).

**Q6. Explain virtual memory**

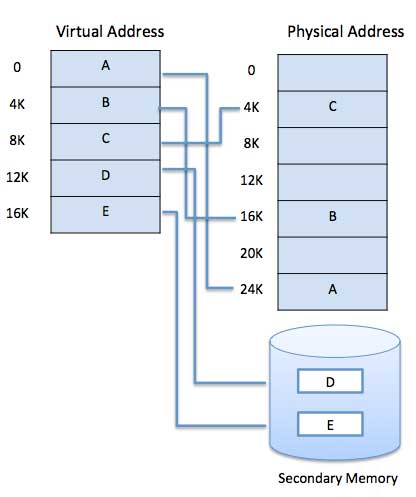
A computer can address more memory than the amount physically installed on the system. This extra memory is actually called **virtual memory** and it is a section of a hard disk that's set up to emulate the computer's RAM.

The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory serves two purposes. First, it allows us to extend the use of physical memory by using disk. Second, it allows us to have memory protection, because each virtual address is translated to a physical address.

Following are the situations, when entire program is not required to be loaded fully in main memory.

* User written error handling routines are used only when an error occurred in the data or computation.
* Certain options and features of a program may be used rarely.
* Many tables are assigned a fixed amount of address space even though only a small amount of the table is actually used.
* The ability to execute a program that is only partially in memory would counter many benefits.
* Less number of I/O would be needed to load or swap each user program into memory.
* A program would no longer be constrained by the amount of physical memory that is available.
* Each user program could take less physical memory, more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.

Modern microprocessors intended for general-purpose use, a memory management unit, or MMU, is built into the hardware. The MMU's job is to translate virtual addresses into physical addresses. A basic example is given below −



Virtual memory is commonly implemented by demand paging. It can also be implemented in a segmentation system. Demand segmentation can also be used to provide virtual memory.

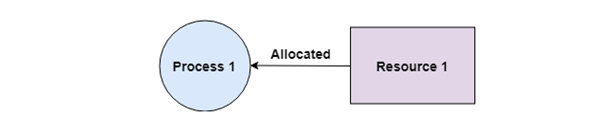
**Q7. Explain Deadlock and its characteristics**

A deadlock happens in operating system when two or more processes need some resource to complete their execution that is held by the other process.

A deadlock occurs if the four Coffman conditions hold true. But these conditions are not mutually exclusive. They are given as follows −

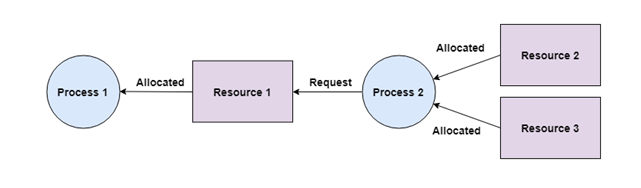
## **Mutual Exclusion**

There should be a resource that can only be held by one process at a time. In the diagram below, there is a single instance of Resource 1 and it is held by Process 1 only.



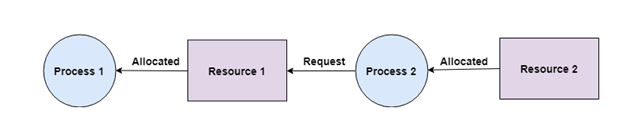
## **Hold and Wait**

A process can hold multiple resources and still request more resources from other processes which are holding them. In the diagram given below, Process 2 holds Resource 2 and Resource 3 and is requesting the Resource 1 which is held by Process 1.



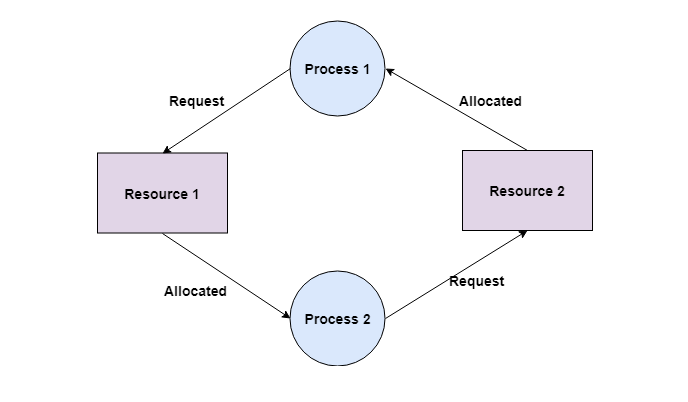
## **No Preemption**

A resource cannot be preempted from a process by force. A process can only release a resource voluntarily. In the diagram below, Process 2 cannot preempt Resource 1 from Process 1. It will only be released when Process 1 relinquishes it voluntarily after its execution is complete.



## **Circular Wait**

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. This forms a circular chain. For example: Process 1 is allocated Resource2 and it is requesting Resource 1. Similarly, Process 2 is allocated Resource 1 and it is requesting Resource 2. This forms a circular wait loop.



**Q8. Explain NAT and ARP protocol ?**

**Network Address Translation (NAT)**

it is designed for IP address conservation. It enables private IP networks that use unregistered IP addresses to connect to the Internet. NAT operates on a router, usually connecting two networks together, and translates the private (not globally unique) addresses in the internal network into legal addresses, before packets are forwarded to another network.

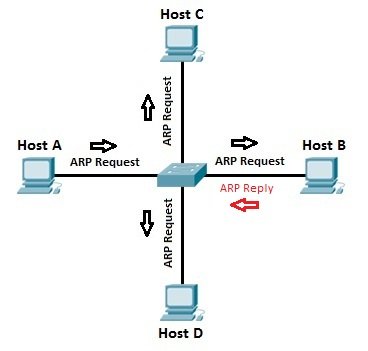
As part of this capability, NAT can be configured to advertise only one address for the entire network to the outside world. This provides additional security by effectively hiding the entire internal network behind that address. NAT offers the dual functions of security and address conservation and is typically implemented in remote-access environments.

# **ARP (Address Resolution Protocol) explained**

**ARP (Address Resolution Protocol)** is a network protocol used to find out the hardware (MAC) address of a device from an IP address. It is used when a device wants to communicate with some other device on a local network (for example on an Ethernet network that requires physical addresses to be known before sending packets). The sending device uses ARP to translate IP addresses to MAC addresses. The device sends an ARP request message containing the IP address of the receiving device. All devices on a local network segment see the message, but only the device that has that IP address responds with the ARP reply message containing its MAC address. The sending device now has enough information to send the packet to the receiving device.

ARP request packets are sent to the broadcast addresses (FF:FF:FF:FF:FF:FF for the Ethernet broadcasts and 255.255.255.255 for the IP broadcast).

Here is the explanation otf the ARP process:

[](https://603168-1953132-raikfcquaxqncofqfm.stackpathdns.com/wp-content/images/arp_process.jpg)

Let’s say that Host A wants to communicate with host B. Host A knows the IP address of host B, but it doesn’t know the host B’s MAC address. In order to find out the MAC address of host B, host A sends an ARP request, listing the host B’s IP address as the destination IP address and the MAC address of FF:FF:FF:FF:FF:FF (Ethernet broadcast). Switch will forward the frame out all interfaces (except the incoming interface). Each device on the segment will receive the packet, but because the destination IP address is host B’s IP address, only host B will reply with the ARP reply packet, listing its MAC address. Host A now has enough information to send the traffic to host B.

All operating systems maintain ARP caches that are checked before sending an ARP request message. Each time a host needs to send a packet to another host on the LAN, it first checks its ARP cache for the correct IP address and matching MAC address. The addresses will stay in the cache for a couple of minutes. You can display ARP entries in Windows by using the arp -a command: