**Ninja Interview Questions**

**Technical**

Technical Round (Core + Basic Coding + Academic Projects)

1. OOP Concepts - Explain with real-life examples (Encapsulation vs Abstraction, Inheritans Polymorphism).

2. DBMS - Difference between DELETE, TRUNCATE, DROP.

3. SQL Query - Find second highest salary / count duplicates / joins (especially INNER JOIN and LEFT JOIN).

4. C / Java / Python Basics - Input/output, conditionals, loops.

5. Data Structures - Difference between array and linked list; basic stack and queue implementation.

6. Sorting - Bubble sort or selection sort implementation.

Array v Linkedlist

Stack v queue

What is os

Count frequency of numner using hashmap

Linux commands (mentioned in CV)

What is Cloud. Types of Cloud (mentioned is resume \_certification)

What your thoughts on AI

Q. List v Touple

Q. What is difference between **bitwise &** operator and **Relation &** Operator

***The bitwise & operator performs an AND operation at the binary level. It compares each bit of two integers and returns a new integer where each bit is set to 1 only if both corresponding bits in the operands are 1.***

***Relational operators, on the other hand, are used to compare values and determine relationships like equality, greater than, or less than. They return Boolean values (True or False) rather than performing binary operations.***

***So, while & works on the binary representation of numbers, relational operators are used for comparisons in control flow and logic.***

**Example:**

# Bitwise AND

5 & 3 # 0101 & 0011 = 0001 → result: 1

# Relational

5 > 3 # result: True

If they also bring up and (logical AND), you can optionally add:

***Additionally, Python has a logical and operator, which is used to combine boolean expressions and short-circuits—meaning it stops evaluating if the result is already known. This is different from &, which always evaluates both sides.***

Q. Bitwise v Logical Operators

In Python, the **bitwise** operators and **logical** operators serve different purposes:

* ***Bitwise operators like &, |, ^, ~, <<, and >> operate at the binary level. They're used to manipulate individual bits of integers.***
* ***Logical operators like and, or, and not work with boolean values (True, False) and are typically used for control flow or conditional logic.***

Q. How can we find 2nd largest Element in array?

***We use sorting the array first with using any of sorting method , or simply sort function in python, then print 2nd last element,***

***I first find the maximum element using Python’s built-in max() function. Then, I scan the list again to find the largest element that is not equal to the maximum — effectively giving the second largest. This avoids sorting and handles duplicates properly.***

Q. **What is a Pure Virtual Function?**

***A pure virtual function is a virtual function in a base class that has no implementation in that class and must be overridden by any derived class. It essentially defines an interface for derived classes.***

***Why use it?***

* ***To create abstract classes — classes that cannot be instantiated directly.***
* ***To ensure that derived classes implement specific functionality.***
* ***Helps in polymorphism by providing a common interface.***

Q. Why is multiple inheritance not allowed in Java?

Java **does not allow multiple inheritance with classes** to avoid the **“Diamond Problem”** — a classic ambiguity problem where a class inherits from two classes that have a method with the same signature.

**What is the Diamond Problem?**

Imagine:

A

/ \

B C

\ /

D

* If both **B** and **C** have a method show(), and **D** inherits from both, the compiler doesn’t know which show() method to use.

This ambiguity can cause **confusion and errors** in the program.

***Java does not support multiple inheritance with classes to prevent ambiguity and complexity caused by the Diamond Problem. Instead, Java uses interfaces to provide multiple inheritance of type, allowing a class to implement multiple interfaces and thus multiple types safely.***

Q. What is hoisting in JavaScript

***Hoisting is JavaScript’s behavior of moving variable and function declarations to the top of their scope before execution. var and function declarations are hoisted, but let, const, and function expressions are not hoisted the same way. This can lead to unexpected behavior if not properly understood.***

***Ex –***

***console.log(x); // Output: undefined***

***var x = 5;***

***var x;***

***console.log(x); // undefined***

***x = 5;***

***The var x declaration is hoisted, but x = 5 is not.***

***console.log(a); // ReferenceError: Cannot access 'a' before initialization***

***let a = 10;***

***let and const are hoisted, but they are in a "temporal dead zone" — you cannot access them before the line where they are declared.***

***Function Hoisting***

***greet(); // Output: Hello!***

***function greet() {***

***console.log("Hello!");***

***}***

***Full function declarations are hoisted with their definitions.***

Q. What is Implicit Type Conversion (Type Coercion) in JavaScript?

***Implicit type conversion, also called type coercion, is when JavaScript automatically converts data types during operations involving different types — without you explicitly asking it to.***

***Ex-***

***let result = "5" + 2;***

***console.log(result); // "52"***

***Implicit type conversion in JavaScript is when the engine automatically converts data types during operations, like turning strings into numbers or booleans into numbers. While convenient, it can cause confusing or unexpected behavior, so it’s important to know when and how coercion happens.***

* ***Explicit Type Conversion/ Type Casting***

***Number("456"); // 456***

***parseInt("456px"); // 456***

***parseFloat("3.14"); // 3.14***

Q. What is the <a> (Anchor) Tag in HTML?

***he <a> tag (short for anchor) is used in HTML to create hyperlinks — clickable links that navigate to another page, file, or location on the same page.***

Ex –

**<a href="https://www.example.com">Visit Example</a>**

**Managerial**

1.How do you handle tight deadlines?

2.Give an example of when you worked in a team.

3.Suppose you're assigned a new technology. How will you learn it quickly?

4. How do you manage college and coding together?

5. How do you prioritize between work and personal life?

**HR**

1. Tell me about yourself.

2. Why TCS? \*

Where do you see yourself in next 5 years

Why not post graduation

Hobbies

Family

3. Are you open to relocation?

4. What are your strengths and weaknesses?

5. Where do you see yourself in 5 years?

6. Are you planning for higher studies?

7. Do you have any backlogs? Explain if yes.