Interview Questions (1 to 183)

# Question 1

Q: OOPs uses & advantages?

A: OOP provides a clear modular structure for programs. It makes development and maintenance easier, supports reusability through inheritance, and enhances scalability with polymorphism.

Example:  
Encapsulation: Wrapping data and methods together.  
Inheritance: A class inherits properties from another class.

Explanation:  
OOP allows modeling real-world objects, making it easier to manage complex programs and enhancing code reusability.

# Question 2

Q: Abstraction vs Inheritance?

A: Abstraction hides implementation details and exposes only essential features. Inheritance allows one class to inherit properties and methods from another.

Example:  
Abstraction Example: Interface IVehicle only exposes Start() method.  
Inheritance Example: Class Car inherits from Vehicle class.

Explanation:  
Abstraction helps in focusing on what an object does instead of how it does it. Inheritance allows code reuse by deriving new classes from existing ones.

# Question 3

Q: Encapsulation and its uses?

A: Encapsulation wraps data (variables) and code (methods) into a single unit. It also restricts access to certain details of an object by using access specifiers.

Example:  
public class House {  
private int rooms;  
public int GetRooms() { return rooms; }  
}

Explanation:  
Encapsulation helps in maintaining control over data and improves security by providing controlled access via methods.

# Question 4

Q: Polymorphism and its uses?

A: Polymorphism allows objects to be treated as instances of their parent class. It enables a single method to operate on different data types.

Example:  
Method Overloading: void Draw(Circle c), void Draw(Rectangle r).  
Method Overriding: Child class overriding a method from a parent class.

Explanation:  
Polymorphism promotes flexibility and scalability in code. It allows a single interface to represent different underlying forms.

# Question 5

Q: Types of Joins and uses?

A: Inner Join: Returns records that have matching values in both tables.  
Left Join: Returns all records from the left table and matched records from the right.  
Right Join: Returns all records from the right table and matched records from the left.  
Full Join: Returns all records when there is a match in either left or right table.

Example:  
SQL Example: SELECT \* FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

Explanation:  
Joins are used to combine rows from two or more tables based on a related column. This is essential for retrieving meaningful data across related datasets.

# Question 6

Q: MVC Cycle

A: MVC stands for Model-View-Controller. It is a software architectural pattern. The Model represents the data, the View displays the data, and the Controller handles user interaction.

Example:  
MVC Process:  
1. User interacts with View.  
2. Controller handles input.  
3. Controller updates Model.  
4. View displays Model data.

Explanation:  
MVC helps in the separation of concerns, making the application more modular, maintainable, and scalable.

# Question 7

Q: Write a program using delegate to display the number in triangular form?

A: A delegate in C# is a type that represents references to methods with a specific parameter list and return type. It can be used to pass methods as arguments to other methods.

Example:  
delegate void PrintTriangle(int n);  
class Program {  
static void Main() {  
PrintTriangle triangle = new PrintTriangle(Triangle);  
triangle(5);  
}  
static void Triangle(int n) {  
for (int i = 1; i <= n; i++) {  
for (int j = 1; j <= i; j++)  
Console.Write('\* ');  
Console.WriteLine();  
}  
}  
}

Explanation:  
The delegate PrintTriangle holds the reference of the Triangle method. This example prints a triangular pattern using nested loops.