# LAB ON C# PRACTICALS LIST

1. Write a C# Sharp program that takes a number as input and print its multiplication table.
2. Write a C# Sharp program that takes four numbers as input to calculate and print the average and product.
3. If you have two integers stored in variables var1 and var2, what Boolean test can you perform to see if one or the other (but not both) is greater than 10?
4. Write an application that receives the following information from a set of students: Student Id: Student Name: Course Name: Date of Birth: The application should also display the information of all the students once the data is entered. Implement this using an Array of Structures.
5. Write a C# Sharp program that takes a number and a width also a number, as input and then displays a triangle of that width, using that number.
6. Write a C# Sharp program to check whether entered number is even or odd.
7. Write a Program in C# to demonstrate Command line arguments Processing.
8. Write a Program in C# to find the roots of Quadratic Equation.
9. Write a Program in C# to demonstrate boxing and Unboxing.
10. Write a Program in C# to find difference between two dates.
11. WAP in C# to display dates in various formats.
12. Write a C# Sharp program to find whether a given year is a leap year or not.
13. Find the sum of all the elements present in a jagged array of 3 in n arrays.
14. Design a simple calculator using Switch Statement in C#.
15. Write a Program in C# to multiply to matrices using Rectangular arrays.
16. Write a program to reverse a given string using C#.
17. WAP in C# to get a number and display the sum of its digits.
18. WAP in C# to get a number and display the number with its reverse, check whether a number is Palindrome or not.
19. WAP concatenate two String, find sub-String and replace sub-String.
20. WAP in C# Sharp to find the sum of all elements of the array.
21. Write a program in C# to demonstrate different types of Constructors.
22. WAP in C# to demonstrate method overloading and overriding.
23. Write a program in C# to demonstrate Operator overloading.
24. Write a program to demonstrate abstract class and abstract methods in C#.
25. Write a program in C# to build a class which implements an interface which already exists.
26. To develop a C# program to implement threading concepts.
27. To develop a C# program to implement the following concepts: (a) Delegates (b) Events
28. WAP in C# to show inheritance (Multilevel and Hierarchical).
29. WAP in C# to demonstrate Destructor.
30. Demonstrate arrays of interface types with a C# program.
31. Write a program to illustrate the use of properties and indexers
32. WAP in C# to design a student registration form that includes window form controls.
33. Consider the Database STUDENT consisting of following tables: Course (C_ID: int, C_Name: string)

    Student (RollNo:int, S_ Name: string, Address: string, C_ID: int, Admissionyear: int)

    Develop suitable windows application using C#.NET having following options:
    - Entering new course details.

- Entering new student details.
- Display the details of students (in a Grid) who belong to a particular course.
- Display the details of students (in a Grid) who have taken admission in a particular year.

34. WAP in C# On Divide by Zero error.
35. WAP in C# On IndexOutOfRangeException.

36. designing a basic payment system for an e-commerce application that supports multiple payment methods, such as credit card payments, PayPal, and bank transfers. Each payment method has some common features but also unique implementations.

1. Create an interface called IPaymentMethod that includes the following methods:
   - bool ValidatePaymentDetails();
   - void ProcessPayment();
   - string GetPaymentMethodName();

2. Create an abstract class called PaymentMethodBase that implements the IPaymentMethod interface.
   - Implement the GetPaymentMethodName method in this class to return the name of the payment method.
   - Declare a protected field paymentMethodName to store the payment method name.
   - Provide a constructor in PaymentMethodBase that accepts a string parameter to initialize the paymentMethodName.
   - Define an abstract method CalculateTransactionFees() in this class that must be implemented by derived classes.

3. Create three concrete classes CreditCardPayment, PayPalPayment, and BankTransferPayment that inherit from PaymentMethodBase.
   - Implement the ValidatePaymentDetails, ProcessPayment, and CalculateTransactionFees methods in each class with logic appropriate to each payment type.

4. Write a simple Main method to demonstrate:
   - Creating instances of each payment type.
   - Calling the ProcessPayment method on each.
   - Displaying the payment method name and the calculated transaction fees

37. Create a Base Class Vehicle:
    1. - Define a base class Vehicle with the following properties:
       - Make (string)
       - Model (string)
       - Year (int)
    - Implement a constructor that initializes these properties.
    - Add a static field TotalVehicles to keep track of the number of Vehicle objects created.
    - Implement a destructor that decrements TotalVehicles when a Vehicle object is destroyed.

    2. Derive a Class Truck:
       - Derive a class Truck from Vehicle with an additional property:

- PayloadCapacity (double)
- Implement a constructor for Truck that initializes all properties, including those inherited from Vehicle, using constructor chaining.
- Add a static field TotalTrucks to keep track of the number of Truck objects created.
- Implement a destructor that decrements TotalTrucks when a Truck object is destroyed.

3. Implement Additional Functionality:
- Add a static method ShowVehicleStatistics() in the Vehicle class that displays the total number of vehicles and trucks created.
- Add a method DisplayInfo() in the Vehicle class to output the details of the vehicle.
- Override DisplayInfo() in the Truck class to include PayloadCapacity in the output.

4. Demonstrate Usage:
- Create a list of Vehicle objects, including Truck objects.
- Display information about each vehicle using the DisplayInfo() method.
- Use the ShowVehicleStatistics() method to display the total number of vehicles and trucks.

38. Write a C# Sharp program to find the factorial of a given number using recursion.
- This program should prompt the user to enter a number and then calculate and display the factorial of that number using a recursive function.

39. Create a C# program to read a list of names from the user and sort them in alphabetical order.
- Implement this using an array or a list, and ensure the names are case-insensitive during sorting.

40. Write a C# Sharp program to demonstrate the concept of exception handling by creating a custom exception class.
- Create a custom exception called InvalidAgeException that is thrown when the user inputs an invalid age (e.g., negative age).

41. Develop a C# program to implement a simple banking application.
- The program should allow the user to create a bank account, deposit money, withdraw money, and check the balance. Use classes to represent the account and manage transactions.

42 Write a C# Sharp program that implements a simple file handling operation.
- The program should create a file, write some text into it, and then read the content from the file and display it on the console.