1. Create a list of integers (`List<int>`) ranging from 1 to 10.

Use a lambda expression to:

- Filter out odd numbers.
- Multiply each number by 2.
- Find the sum of all the even numbers. _Hitarth

2. Create a delegate that takes two integers and performs a calculation (add, subtract, or multiply).
- Write a program where a user selects the operation, and the delegate invokes the corresponding method. -Rajvi

3. Create an extension method for the `string` class that counts the number of vowels in a given string.method allow the user to specify which characters should be counted as vowels. For example, the user could provide a list of characters (not just a, e, i, o, u) to count as "vowels". -Ishika

4. You are working in the HR department of a company, and your task is to manage employee records. Each employee has the following properties: **Name**, **Age**, **Department**, and **Salary**. You need to perform the following tasks using **lambda expressions**:

   Filter the employees who are older than 30 years.

   Sort the employees by their **Salary** in descending order.

   Transform the list of employees into a list of employee names who are working in the "Sales" department.

   (Use the lambda expression) -Fatema

5. Create a C# program that demonstrates the use of **extension methods** to enhance built-in types like `string` and `int`. The program will not only implement common string and numeric operations using extension methods but also integrate some extra functionality that makes the extension methods more flexible and dynamic.

Scenario:

You are working on a custom logging utility for a web application. The utility needs to:

1. **Add a timestamp** to each log message (using `string` extension).
2. **Filter log messages** based on whether they contain certain keywords (using `string` extension).
3. **Log even numbers** and **calculate the sum of odd numbers** (using `int` extension). -

   -Shubh

6. Create a generic class called `GenericList<T>` that can store a collection of elements of any type. The class should allow the following operations:

   Add an element to the list.

   Get the element at a specific index.

   Remove an element by its value.

   Print all elements in the list.

   Then, demonstrate its usage by creating instances of `GenericList<T>` with different types (e.g., `int`, `string`, `Student`). -Mokshang

7. Create a generic method `SortList<T>` that sorts a list of elements. This method should work for any type that implements `IComparable<T>`. (don't use the inbuilt method for sorting) -Vasu

8. Create a set of extension methods for the `string` type that perform common string manipulations. The task is to create the following methods as extension methods:

   `ToTitleCase()`: Converts a string to title case (capitalizes the first letter of each word).

   `ReverseString()`: Reverses the characters of the string.

   `IsPalindrome()`: Checks if the string is a palindrome (reads the same forwards and backwards).

`WordCount()`: Counts the number of words in the string (words are separated by spaces). -Meghal

9. Create a `List<T>` of value types, then **box each value** and store it in a `List<object>`.

   The program should not only box and unbox a single value type but also use **collections** (like `List<T>` or `ArrayList`) to store both boxed and unboxed values.

   Also use **generic method** that works with any value type (like `int`, `double`, or `char`) and boxes and unboxes the value -Dhirav

10. Create a C# program that demonstrates the use of **delegates** and **anonymous methods**. The program should simulate a scenario where different types of operations (arithmetic calculations, string manipulations) are performed using delegates, with the operations defined as anonymous methods. - Utsav

11. Create a C# program to simulate a **bank account system** where various actions (like depositing money, withdrawing money) trigger events. The program will use **events** and **event handlers** to respond to these actions, and multiple event handlers will take different actions when the events are raised. -Devansh