Module 6

What to learn

- LINQ
- LINQ Query
- LINQ Methods
- Query Operators
    - Where
    - OfType
    - OrderBy
    - GroupBy,ToLookup
    - Join
    - GroupJoin

Practice Exercise

Practice 1

- Write a LINQ query to find all even numbers from a list of integers and display them in ascending order.
- Create a LINQ query to calculate the average salary of employees whose job title is "Manager".
- Using LINQ, retrieve the top 3 highest-scoring students from a list of scores.
- Implement a LINQ query to find the distinct departments in a company database.
- Use LINQ to create a new list of objects by projecting fields from an existing list (e.g., extract FirstName and LastName into a FullName property).

Practice 2

- Write a LINQ query to retrieve employees whose age is between 25 and 35, sorted by their names.
- Using LINQ, find all products that are out of stock from a product list.
- Implement a query to group a collection of words by their first letter and count how many words start with each letter.
- Create a LINQ query to find customers who placed orders in the last month, sorted by order date.
- Write a query to calculate the total quantity of each product sold from an order detail collection.

Practice 3

LINQ Methods

- Use Select to transform a list of employee objects into a list of their email addresses.
- Use FirstOrDefault to find the first student with a grade above 90, or return null if none exist.
- Implement a LINQ chain using Select and Where to find the names of people aged above 30.
- Use Aggregate to find the concatenated string of all book titles in a collection, separated by commas.

Use Take and Skip to implement pagination for displaying 5 records per page from a large dataset.

Practice 4

Query Operators

Where

Write a query to filter all employees with a salary greater than $50,000.
Use Where to retrieve all even numbers from a list of integers.
Write a LINQ query to find all products that are both in stock and cost less than $20.
Use Where to filter students whose names start with a specific letter provided by the user.
Retrieve all records from a dataset where a specific field (e.g., age) satisfies a dynamic condition.

Practice 5

OfType

Use OfType<T> to filter only integer elements from a mixed collection.
Write a query to find all strings from an ArrayList of mixed data types.
Use OfType to filter all floating-point numbers from a collection and calculate their average.
Implement a LINQ query to extract all objects of a specific type (e.g., Product) from a collection of objects.
Retrieve only DateTime elements from a list containing various data types.

Practice 6

OrderBy

Write a query to sort a list of students by their grades in descending order.
Use OrderBy to arrange employee names alphabetically.
Sort a list of products first by category and then by price within each category using OrderBy and ThenBy.
Implement a LINQ query to sort a collection of strings by their length.
Write a LINQ query to sort a dataset by multiple fields, dynamically chosen at runtime.

Practice 7

GroupBy, ToLookup

Use GroupBy to group a list of employees by their department and count the number of employees in each department.
Write a LINQ query to group a collection of products by their categories and calculate the total stock for each category.
Implement a query using ToLookup to group customers by their country and retrieve all customers from a specific country.
Use GroupBy to group a list of words by their length and sort the groups by the length of the words.
Implement a LINQ query to group orders by their status and find the total revenue for each status.

Practice 8

Join

Write a LINQ query to join two lists: employees and departments, and display employee names along with their department names.

Use Join to combine a list of orders with a list of customers based on customer ID.
Implement a query to join a list of students with their grades and display the student's name and grade.
Use Join to create a combined list of products and their suppliers based on a common field.
Write a LINQ query to join two lists and filter the result by a specific condition (e.g., orders placed after a specific date).

Practice 9

GroupJoin

Use GroupJoin to group products by their categories and display the category name along with its products.
Write a query to group customers with their orders using GroupJoin, displaying the customer name and their orders.
Implement a LINQ query to group a list of teachers with their students based on a TeacherId.
Use GroupJoin to combine employees with their projects and calculate the number of projects each employee is handling.
Write a LINQ query to group departments with their employees and sort the departments by the number of employees.

Assignment Exercise

Assignment 1

Create a Two Classes Employees and Incentives class Employee { public int ID { get; set; } public string FirstName { get; set; } public string LastName { get; set; } public double Salary { get; set; } public DateTime JoiningDate { get; set; } public string Deparment { get; set; } } class Incentive { public int ID { get; set; } public double IncentiveAmount { get; set; } public DateTime IncentiveDate { get; set; } } class Program { static void Main(string[] args) { List employees = new List() { new Employee(){ ID=1,FirstName="John",LastName="Abraham",Salary=1000000,JoiningDate=new DateTime(2013,1,1),Deparment="Banking"}, new Employee() {ID=2,FirstName="Michael",LastName="Clarke",Salary=800000,JoiningDate=new DateTime(),Deparment="Insurance" }, new Employee() {ID=3,FirstName="oy",LastName="Thomas",Salary=700000,JoiningDate=new DateTime() ,Deparment="Insurance"}, new Employee() {ID=4,FirstName="Tom",LastName="Jose",Salary=600000,JoiningDate=new DateTime() ,Deparment="Banking"}, new Employee() {ID=4,FirstName="TestName2",LastName="Lname%",Salary=600000,JoiningDate=new DateTime(2013,1,1) ,Deparment="Services"} }; List incentives = new List() { new Incentive(){ ID=1,IncentiveAmount=5000,IncentiveDate=new DateTime(2013,02,02)}, new Incentive() {ID=2,IncentiveAmount=10000,IncentiveDate=new DateTime(2013,02,4)}, new Incentive() {ID=1,IncentiveAmount=6000,IncentiveDate=new DateTime(2012,01,5)}, new Incentive() {ID=2,IncentiveAmount=3000,IncentiveDate=new DateTime(2012,01,5)} }; } } Solve below queries:

Assignment 2

Get employee details from employees object whose employee name is "John" (note restrict operator)

Assignment 3

Get FIRSTNAME,LASTNAMe from employees object(note project operator)

Assignment 4

Select FirstName, IncentiveAmount from employees and incentives object for those employees who have incentives.(join operator)

Assignment 5

Get department wise maximum salary from employee table order by salary ascending (note group by)

Assignment 6

Select department, total salary with respect to a department from employees object where total salary greater than 800000 order by TotalSalary descending(group by having)

Assignment 7

Scenario

You are tasked with developing a **Student Management System** for a college. The system will manage data about students, courses, and enrollments. It will allow administrators to query and analyze data using LINQ.

Requirements

Data Models

Create the following classes:

Student

- StudentId (int)
- Name (string)
- Age (int)
- Gender (string)
- Department (string)

Course

- CourseId (int)
- CourseName (string)
- Department (string)

Enrollment

- EnrollmentId (int)
- StudentId (int)
- CourseId (int)
- EnrollmentDate (DateTime)

Tasks

Task 1: Filter and Analyze Data (Where and OfType)

- Retrieve all students who belong to the "Computer Science" department and are older than 20 years.
- Filter and display all course names from a mixed collection of objects using OfType<string>.

Task 2: Sorting and Grouping (OrderBy, GroupBy, ToLookup)

Sort all students by their name in ascending order. Then, sort them by age in descending order.
Group all students by their department and display the total count of students in each department.
Use ToLookup to group courses by department and retrieve all courses belonging to a specific department (e.g., "Mathematics").

Task 3: Joining Data (Join and GroupJoin)

Write a LINQ query to join students with their enrollments and display the student name along with the course name they are enrolled in.
Use GroupJoin to group students with the courses they are enrolled in and display the student name along with the list of course names.

Task 4: Advanced Queries (LINQ Query and Methods)

Calculate the total number of enrollments in each department and display the department name along with the count.
Find the student(s) with the maximum number of enrollments.
Write a LINQ query to find all courses that have no enrollments.

Student Management System
---
1. View Students in a Department
2. View Courses by Department
3. View Enrollments (Student and Course)
4. Find Students with Maximum Enrollments
5. View Courses with No Enrollments
6. Exit

Online Reference

https://www.tutorialsteacher.com/linq/linq-lambda-expression

.NET Core Web API

WEB API (old)

Authentication And Authorization (WEBAPI)(old)

FullStackDevelopment_WIth_Dotnet_AND_Angular