Práctica 2 – LinQ desde listas.

Ejercicio 1

Se requiere que se realicen dos consultas adicionales utilizando LinQ, basadas en el siguiente ejemplo:

i. Mostrar el nombre de todas las personas que tengan una edad mayor a 30 y vivan en Bogotá.
ii. Mostrar el nombre de todas las personas que tengan una edad entre 25 y 35 años, ordenadas por edad de forma ascendente.

Suba el código a un repositorio remoto, para compartirlo con el docente.

```
internal class Program
{
    0 referencias
    static void Main(string[] args)
    {
        List<Persona> personas = new List<Persona>
            {
                new Persona { Nombre = "Juan", Edad = 25, Ciudad = "Lima" },
                new Persona { Nombre = "María", Edad = 30, Ciudad = "Bogotá" },
                new Persona { Nombre = "Pedro", Edad = 35, Ciudad = "Madrid" },
                new Persona { Nombre = "Ana", Edad = 20, Ciudad = "Lima" },
                new Persona { Nombre = "José", Edad = 40, Ciudad = "Buenos Aires" }
            };

        var consulta = from p in personas where p.Edad < 25 && p.Ciudad == "Lima"
                       orderby p.Nombre descending select new { p.Nombre, p.Edad };

        foreach (var persona in consulta)
        {
            Console.WriteLine($"{persona.Nombre} ({persona.Edad} años)");
        }
    }
}

internal class Persona
{
    7 referencias
    public int Edad { get; set; }
    7 referencias
    public string Nombre { get; set; }
    6 referencias
    public string Ciudad { get; set; }
}
```

Ejercicio 2

Use el siguiente ejemplo para crear tres empresas distintas, con sus respectivas nóminas y sueldos. Luego suba el código a un repositorio remoto, para compartirlo con el docente.

```
internal class Empleado
{
    9 referencias
    public int Id { get; set; }
    10 referencias
    public string Nombre { get; set; }
    10 referencias
    public string Cargo { get; set; }
    11 referencias
    public int Salario { get; set; }
    11 referencias
    public int EmpresaId { get; set; }
    4 referencias
    public void getDatosEmpleado()

    {
        Console.WriteLine("Empleado {0} con Id {1}, con cargo {2}, con salario {3} y pertenece a " +
            "la empresa {4}", Nombre, Id, Cargo, Salario, EmpresaId);
    }
}
```

```
internal class ControlEmpresasEmpleados
{
    public List<Empresa> listaEmpresas;
    public List<Empleado> listaEmpleados;
    1 referencia
    public ControlEmpresasEmpleados()
    {
        listaEmpresas = new List<Empresa>();
        listaEmpleados = new List<Empleado>();

        listaEmpresas.Add(new Empresa { Id = 1, Nombre = "IAlpha" });
        listaEmpresas.Add(new Empresa { Id = 2, Nombre = "Udelar" });
        listaEmpresas.Add(new Empresa { Id = 3, Nombre = "SpaceZ" });

        listaEmpleados.Add(new Empleado { Id = 1, Nombre = "Gonzalo", Cargo = "CEO", EmpresaId = 1, Salario = 3000 });
        listaEmpleados.Add(new Empleado { Id = 2, Nombre = "JuanC", Cargo = "Desarrollador", EmpresaId = 1, Salario = 2000 });
        listaEmpleados.Add(new Empleado { Id = 3, Nombre = "JuanR", Cargo = "Desarrollador", EmpresaId = 1, Salario = 2000 });
        listaEmpleados.Add(new Empleado { Id = 4, Nombre = "Daniel", Cargo = "Desarrollador", EmpresaId = 1, Salario = 2000 });
        listaEmpleados.Add(new Empleado { Id = 5, Nombre = "GonzaloT", Cargo = "CEO", EmpresaId = 2, Salario = 2000 });
        listaEmpleados.Add(new Empleado { Id = 6, Nombre = "Leonardo", Cargo = "CEO", EmpresaId = 1, Salario = 3000 });
        listaEmpleados.Add(new Empleado { Id = 1, Nombre = "Gonzalo", Cargo = "CEO", EmpresaId = 3, Salario = 3000 });
        listaEmpleados.Add(new Empleado { Id = 6, Nombre = "Leonardo", Cargo = "CEO", EmpresaId = 3, Salario = 3000 });

    }
    //_______Getters_____________________________
    1 referencia
    public void getSeo(string _Cargo)
    {
        IEnumerable<Empleado> empleados = from empleado in listaEmpleados
                                          where empleado.Cargo == _Cargo
                                          select empleado;
        foreach (Empleado elemento in empleados)
        {
            elemento.getDatosEmpleado();
        }
    }
    1 referencia
    public void getEmpleadosOrdenados()
    {
        IEnumerable<Empleado> empleados = from empleado in listaEmpleados
                                          orderby empleado.Nombre
                                          select empleado;
        foreach (Empleado elemento in empleados)
        {
            elemento.getDatosEmpleado();
        }
    }
    1 referencia
    public void getEmpleadosOrdenadosSegun()
    {
        IEnumerable<Empleado> empleados = from empleado in listaEmpleados
                                          orderby empleado.Salario
                                          select empleado;
        foreach (Empleado elemento in empleados)
        {
            elemento.getDatosEmpleado();
        }
    }
    1 referencia
    public void getEmpleadosEmpresa(int _Empresa)
    {
        IEnumerable<Empleado> empleados = from empleado in listaEmpleados
                                          join empresa in listaEmpresas on empleado.EmpresaId
                                          equals empresa.Id
                                          where empresa.Id == _Empresa
                                          select empleado;
        foreach (Empleado elemento in empleados)
        {
            elemento.getDatosEmpleado();
        }
    }

    //_________Metodos Particulares_____________
    1 referencia
    public void pormedioSalario()
    {
        var consulta = from e in listaEmpleados
                       group e by e.EmpresaId into g
                       select new { empresa = g.Key, PromedioSalario = g.Average(e => e.Salario) };

        foreach (var resultado in consulta)
        {
            switch (resultado.empresa)
            {
                case 1: Console.WriteLine($"Empresa IAlpha - Promedio de salario: {resultado.PromedioSalario}");
                break;

                case 2: Console.WriteLine($"Empresa UdeLaR- Promedio de salario: {resultado.PromedioSalario}");
                break;

                case 3: Console.WriteLine($"Empresa SpaceZ - Promedio de salario: {resultado.PromedioSalario}");
                break;
            }

        }
    }
}
```

```
internal class Empresa
{
    6 referencias
    public int Id { get; set; }
    4 referencias
    public string Nombre { get; set; }
    0 referencias
    public void getDatoEmpresa()
    {
        Console.WriteLine("Empresa {0} con Id {1}", Nombre, Id);
    }
}
```

```
internal class Program
{
    0 referencias
    static void Main(string[] args)
    {
        ControlEmpresasEmpleados ce = new ControlEmpresasEmpleados();

        Console.WriteLine("Promedios por empresas \n**************************");
        ce.pormedioSalario();
        Console.WriteLine("");

        Console.WriteLine("Peces Gordos \n**************************");
        ce.getSeo("CEO");

        Console.WriteLine("");
        Console.WriteLine("Plantilla \n**************************");
        ce.getEmpleadosOrdenados();
        Console.WriteLine("");
        Console.WriteLine("Plantilla ordenada por salario \n**************************");
        ce.getEmpleadosOrdenadosSegun();

        Console.WriteLine("\nIngrese la empresa:(entero 1 a 3)\n1 para IAlpha\n2 para UdelaR\n3 para SpaceZ");
        string _Id = Console.ReadLine();
        try
        {
            int _Empresa = int.Parse(_Id);
            ce.getEmpleadosEmpresa(_Empresa);
        }
        catch
        {
            Console.WriteLine("Ha introducido un Id erroneo. Debe ingresar un numero entero");
        }
    }
}
```

Ejercicio 3

Dado el siguiente bloque de código:

i. Calcular la complejidad cognitiva del bloque.
ii. Crear una función que utilice LinQ y muestre en pantalla el mismo resultado.

```
List<int> valores = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

for (var indice = 0; indice < valores.Count - 1; indice++)
{

    if (valores[indice] > valores[indice + 1])
    {
        var valorTemporal = valores[indice];

        valores[indice] = valores[indice + 1];
        valores[indice + 1] = valorTemporal;

        indice = -1;
    }
}

foreach (var valorOrdenado in valores)
{
    Console.WriteLine(valorOrdenado);
}
```

Ejercicio 4

Dados los siguientes bloques de código:

i. Crear una función que utilice LinQ y muestre en pantalla el mismo resultado.

a)

```
List<int> valores = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
int sumaTotal = 0;

foreach (var valor in valores)
{
    sumaTotal = +valor;
}

Console.WriteLine($"La suma total es: {sumaTotal}");
```

b)

```
List<int> valores = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
int sumaTotalValoresPares = 0;

foreach (var valor in valores)
{
    if (valor % 2 == 0)
    {
        sumaTotalValoresPares = +valor;
    }
}

Console.WriteLine($"La suma total de los valores pares es: {sumaTotalValoresPares}");
```

Ejercicio 5

Dado el siguiente bloque de código:

i. Disminuir la complejidad cognitiva del método sin utilizar LinQ.
ii. Crear una función que utilice LinQ y muestre en pantalla el mismo resultado.

```
List<int> valores = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 };

int sumaTotalValoresParesMayoresAOcho = 0;
int sumaTotalValoresParesMenoresAOcho = 0;

foreach (var valor in valores) //+1
{
    if (valor % 2 == 0) //+2
    {
        if (valor > 8) //+3
        {
            sumaTotalValoresParesMayoresAOcho = +valor;
        }
        else //+1
        {
            sumaTotalValoresParesMenoresAOcho = +valor;
        }
    }
}

Console.WriteLine($"La suma total de los valores pares mayores a ocho es: {sumaTotalValoresParesMayoresAOcho}");
Console.WriteLine($"La suma total de los valores pares menores a ocho es: {sumaTotalValoresParesMenoresAOcho}");
```

Ejercicio 6

Dado el siguiente bloque de código:

i. Calcular la complejidad cognitiva del bloque.
ii. Disminuir la complejidad cognitiva del método sin usar LinQ.

```
Console.WriteLine("Ingrese una letra minúscula (desde a hasta f) para saber cual es la siguiente letra del abecedario!!");
string letra = Console.ReadLine();

if(letra == "a")
{
    Console.WriteLine("La siguiente letra del abecedario es B !!");
}

if (letra == "b")
{
    Console.WriteLine("La siguiente letra del abecedario es C !!");
}

if (letra == "c")
{
    Console.WriteLine("La siguiente letra del abecedario es D !!");
}

if (letra == "d")
{
    Console.WriteLine("La siguiente letra del abecedario es E !!");
}

if (letra == "e")
{
    Console.WriteLine("La siguiente letra del abecedario es F !!");
}

if (letra == "f")
{
    Console.WriteLine("La siguiente letra del abecedario es G !!");
}
```