EXAMEN DE REVALIDA

Nombre: Villalta Miranda Italo Harold Materia: SIS2420 “B”

1¿Cuál es la diferencia entre Historia de Usuario y Requerimiento Funcional ?

R.- Una historia de usuario describe que quiere hacer el usuario y porque

El requerimiento funcional detalla exactamente que se debe hacer y como debe hacer el sistema

2¿Qué entiende por paradigma de programación? Que paradigma

recomienda para el proyecto de programación de la materia (justifique

su respuesta)

R.- Un paradigma de programación es un estilo de escribir código que guía cómo se estructuran y resuelven los problemas

Yo recomendaría POO porque el lenguaje que se usó fue c# , por el mantenimiento y escalabilidad y por el modelado

3¿Que se realiza en el grooming y quienes participan?

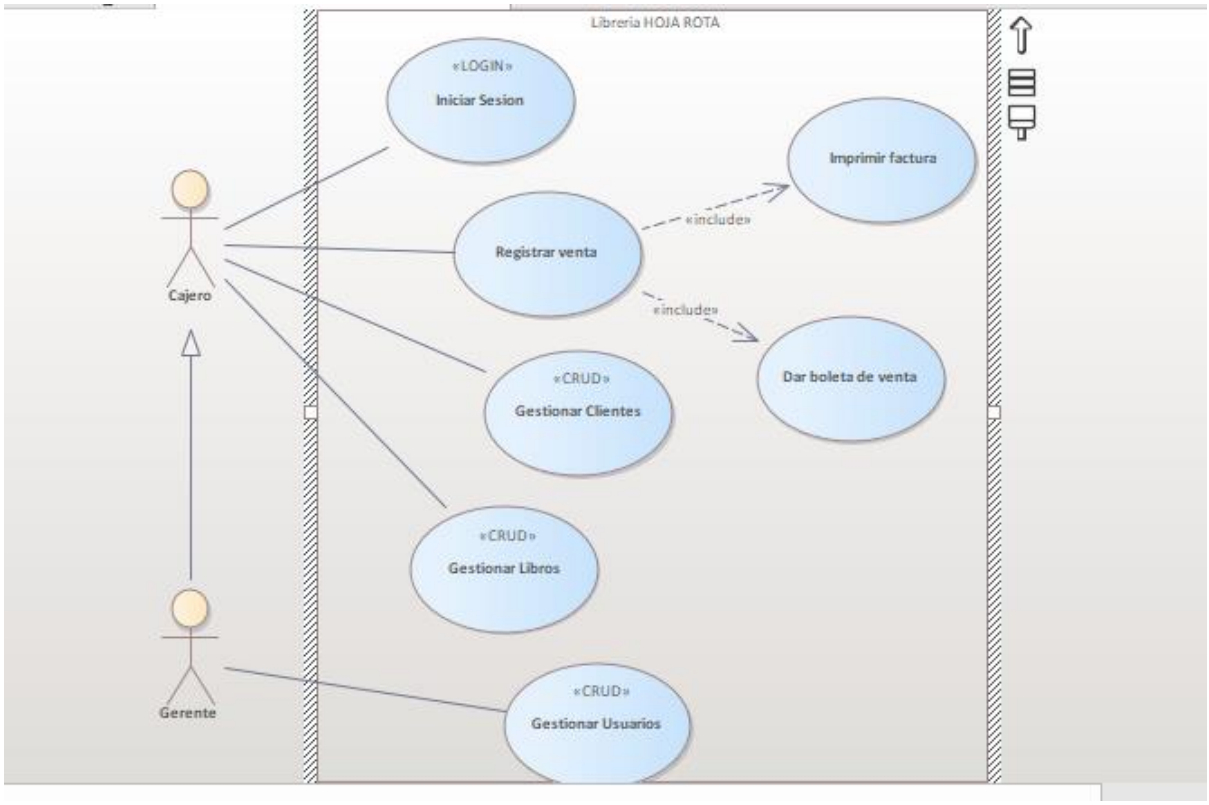
En el grooming, se refinan y priorizan las historias de usuario del backlog para preparar próximas sprints. Participan el Product Owner, el Scrum Master y el equipo de desarrollo.

4 Explique los errores sintácticos y semánticos de los siguientes

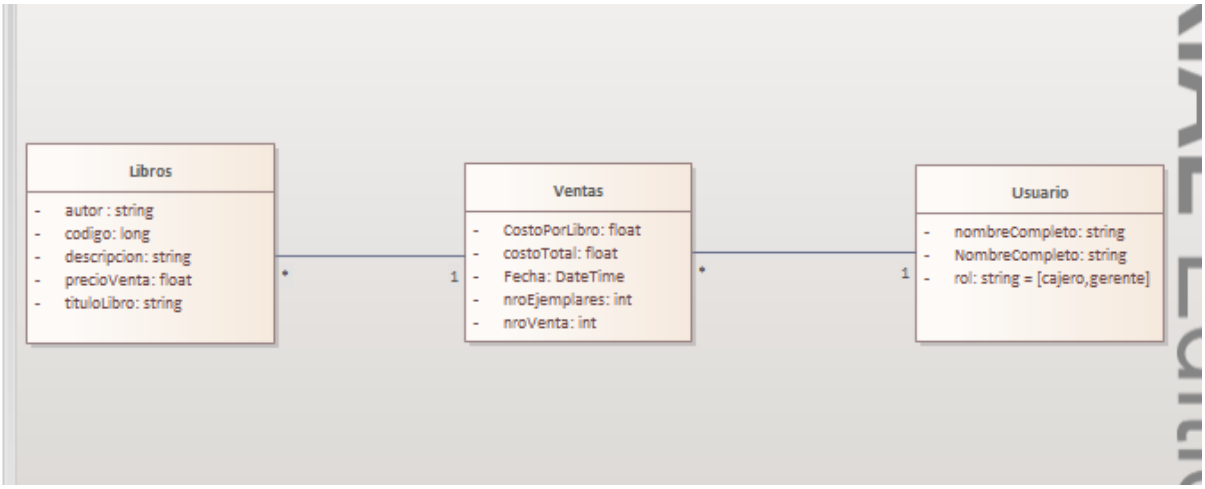
diagramas. Tome en cuenta también las buenas prácticas.

R.-

En el diagrama de casos de uso cambie los conectores de gererente y cajero , cajero y gestionar clientes , los includes de iniciar sesion (login) y asociatividad entre cajero y iniciar sesion, de registro de venta cambien el extend por include y el sentido del include con registro de venta y el nombre de boleto de venta a dar boleto de venta, tambiem gestion de usuarios por Gestionar Usuarios



Del diagrama de casos de usos cambie los tipos de dato de: autor , codigo, precioVenta, costoPorLibro, el de fecha , nombreCompleto, tabien cambie las variables precio_venta por precioVenta, nro_venta por nroVenta y nombre_completo por nombreCompleto tambien la multiplicidad de libros con ventas una venta tiene muchos libros y el conector de Ventas y Usuarios su multiplicidad estaba vbien

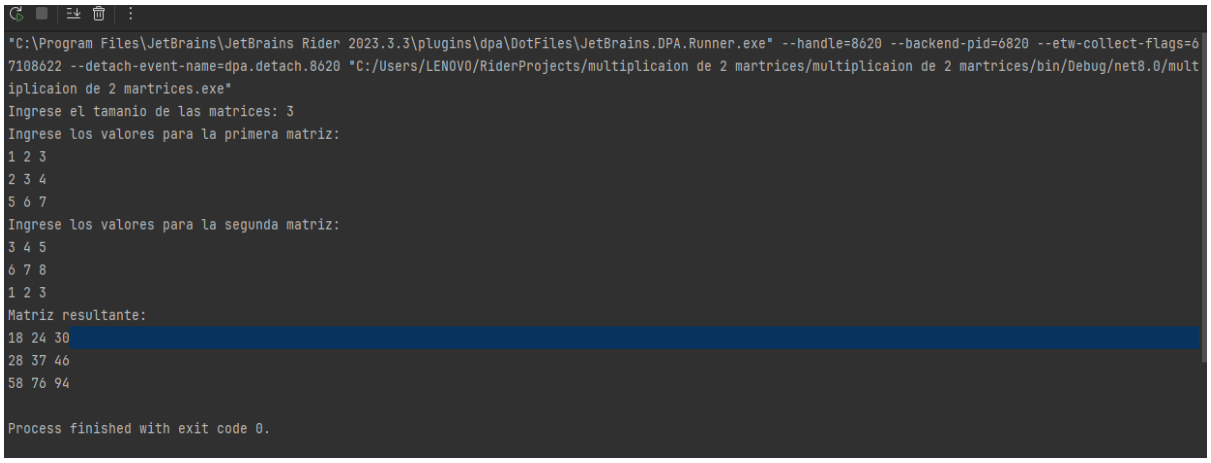


5 programar en c#

Problema de las matrices

```
// See https://aka.ms/new-console-template for more informationusing System;

using System;

class MatrixMultiplication
{
   static void Main()
   {

      Console.Write("Ingrese el tamanio de las matrices: ");
       int size = int.Parse(Console.ReadLine());
       int[,] matrix1 = new int[size, size];
       int[,] matrix2 = new int[size, size];
       int[,] resultMatrix = new int[size, size];
       Console.WriteLine("Ingrese los valores para la primera matriz:");
       for (int i = 0; i < size; i++)
       {
          string[] row = Console.ReadLine().Split(' ');
          for (int j = 0; j < size; j++)
          {
             matrix1[i, j] = int.Parse(row[j]);
          }
       }

      Console.WriteLine("Ingrese los valores para la segunda matriz:");
       for (int i = 0; i < size; i++)
       {
          string[] row = Console.ReadLine().Split(' ');
          for (int j = 0; j < size; j++)
          {
             matrix2[i, j] = int.Parse(row[j]);
          }
       }

      for (int i = 0; i < size; i++)
       {
          for (int j = 0; j < size; j++)
          {
             resultMatrix[i, j] = 0;
             for (int k = 0; k < size; k++)
             {
```

```
                resultMatrix[i, j] += matrix1[i, k] * matrix2[k, j];
            }
        }
    }

    Console.WriteLine("Matriz resultante:");
    for (int i = 0; i < size; i++)
    {
        for (int j = 0; j < size; j++)
        {
            Console.Write(resultMatrix[i, j] + " ");
        }
        Console.WriteLine();
    }
  }
}
```



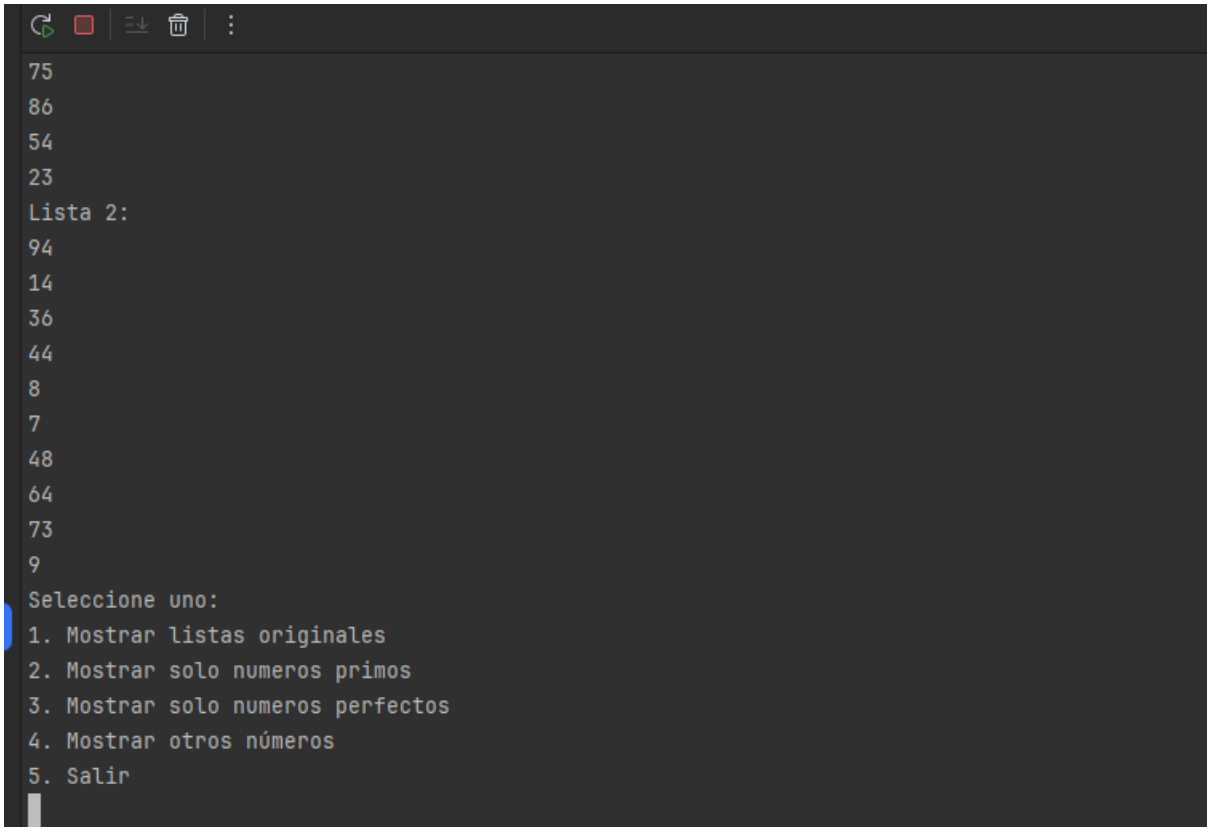
Problema de las listas

```
// See https://aka.ms/new-console-template for more information

using System;
using System.Collections.Generic;
using System.Linq;

class Program
{
  static void Main()
  {
    LinkedList<int> list1 = GenerateRandomList(10, 1, 100);
```

```
LinkedList<int> list2 = GenerateRandomList(10, 1, 100);
LinkedList<int> combina = new LinkedList<int>(list1.Concat(list2));
LinkedList<int> primNum = new LinkedList<int>();
LinkedList<int> perfNum = new LinkedList<int>();
LinkedList<int> otroNum = new LinkedList<int>();
foreach (int num in combina)
{
  if (prim(num))
  {
    primNum.AddLast(num);
  }
  else if (perf(num))
  {
    perfNum.AddLast(num);
  }
  else
  {
    otroNum.AddLast(num);
  }
}
while (true)
{
  Console.WriteLine("Seleccione uno:");
  Console.WriteLine("1. Mostrar listas originales");
  Console.WriteLine("2. Mostrar solo numeros primos");
  Console.WriteLine("3. Mostrar solo numeros perfectos");
  Console.WriteLine("4. Mostrar otros números");
  Console.WriteLine("5. Salir");

  int op = int.Parse(Console.ReadLine());
  switch (op)
  {
    case 1:
      Console.WriteLine("Lista 1:");
      DisplayList(list1);
      Console.WriteLine("Lista 2:");
      DisplayList(list2);
      break;
    case 2:
      Console.WriteLine("Numeros primos:");
      DisplayList(primNum);
```

```
            break;
          case 3:
            Console.WriteLine("Numeros perfectos:");
            DisplayList(perfNum);
            break;
          case 4:
            Console.WriteLine("Otros numeros:");
            DisplayList(otroNum);
            break;
          case 5:
            return;
          default:
            Console.WriteLine("Opcion no valida!! :v  seleccione una opcion del 1 al
5.>:r");
            break;
        }
      }
    }
    static LinkedList<int> GenerateRandomList(int size, int minval, int maxval)
    {
      Random random = new Random();
      LinkedList<int> list = new LinkedList<int>();
      for (int i = 0; i < size; i++)
      {
        list.AddLast(random.Next(minval, maxval + 1));
      }
      return list;
    }
    static bool prim(int num)
    {
      if (num <= 1) return false;
      if (num == 2) return true;
      if (num % 2 == 0) return false;
      for (int i = 3; i <= Math.Sqrt(num); i += 2)
      {
        if (num % i == 0) return false;
      }
      return true;
    }
    static bool perf(int num)
    {
```

```
        int sum = 1;
        for (int i = 2; i <= Math.Sqrt(num); i++)
        {
            if (num % i == 0)
            {
                if (i == num / i)
                    sum += i;
                else
                    sum += i + num / i;
            }
        }
        return sum == num && num != 1;
    }
    static void DisplayList(LinkedList<int> list)
    {
        foreach (int num in list)
        {
            Console.WriteLine(num);
        }
    }
}
```


75
86
54
23
Lista 2:
94
14
36
44
8
7
48
64
73
9
Seleccione uno:
1. Mostrar listas originales
2. Mostrar solo numeros primos
3. Mostrar solo numeros perfectos
4. Mostrar otros números
5. Salir