```
using System;
using System.Collections.Generic;
using System.Security.AccessControl;
using ConsoleApp2;

namespace ConsoleApp2App
{
   internal class Program
   {
      static void Main(string[] args)
      {
         //Criando um onjeto da lista de eventos
         List<Eventos_Filtro> listaEventos = new List<Eventos_Filtro>();
         //Variável para tornar o tratamento de evento iterável
         string opcao;

         do
         {
            Console.Clear();
            Console.WriteLine("=== Gerenciador de Eventos ===");
            Console.WriteLine("1 - Criar novo evento");
            Console.WriteLine("2 - Listar todos os eventos");
            Console.WriteLine("3 - Ordenar eventos");
            Console.WriteLine("4 - Filtrar um evento");
            Console.WriteLine("5 - Excluir evento");
            Console.WriteLine("0 - Sair");
            Console.Write("Escolha uma opção: ");
            opcao = Console.ReadLine();

            switch (opcao)
            {
```

```
            //Criar um evento
            case "1":
               Console.Clear();
               Eventos_Filtro.CriarEvento(listaEventos);
            break;



            //Listar todos os eventos
            case "2":
               Console.Clear();
               //Retorna a seguinte mensagem de erro caso não haja nenhum evento
               if (listaEventos.Count == 0)
               {
                  Console.WriteLine("Nenhum evento cadastrado.");
               }
               else
               {
                  Console.WriteLine("--- Lista de Todos os Eventos ---");
                  Console.WriteLine("Total de eventos cadastrados: " + listaEventos.Count);
                  foreach (var evento in listaEventos) //Imprimindo todos os dados do evento
para cada evento encntrado
                  {
                     Console.WriteLine($"\nID: {evento.ID}");
                     Console.WriteLine($"Nome: {evento.Nome}");
                     Console.WriteLine($"Descrição {evento.Descricao}");
                     Console.WriteLine($"Local: {evento.Local}");
                     Console.WriteLine($"Horário: {evento.Horario}");
                     Console.WriteLine($"Data: {evento.Data.ToShortDateString()}");
                     Console.WriteLine($"Preço: R$ {evento.Preco:F2}");
                  }
               }
```

```
    Console.WriteLine("\nPressione qualquer tecla para continuar...");
    Console.ReadKey();
break;

//Ordenar eventos
case "3":
    Console.Clear();
    Eventos_Filtro.OrdenarEvento(listaEventos);
    Console.WriteLine("\nPressione qualquer tecla para continuar...");
    Console.ReadKey();
break;

//Filtrar eventos
case "4":
    Console.Clear();
    Console.WriteLine("--- Lista de Todos os Eventos ---");
    Console.WriteLine("Total de eventos cadastrados: " + listaEventos.Count);
    Eventos_Filtro.FiltrarEvento(listaEventos);
break;

//Excluir evento
case "5":
    Console.Clear();
    Console.WriteLine("--- Lista de Todos os Eventos ---");
    Console.WriteLine("Total de eventos cadastrados: " + listaEventos.Count);
    Eventos_Filtro.DeletarEvento(listaEventos);
break;

//Fechando o programa
case "0":
    Console.WriteLine("Saindo...");
```

```
                break;

                //Retorna o seguinte erro caso o usuário não digite nenhuma das opções válidas:
                default:
                    Console.WriteLine("Opção inválida! Tente novamente.");
                    Console.ReadKey();
                break;
            }

        } while (opcao != "0"); //Condição para não fechar o console
    }
  }
}
```