File handling:

create a class Network and read data from file and display all the records in below format.

After display:

user can search by id r src r destination r status r network

based on user input records should filtered and display

```
using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;


public class NetworkLog

{

  public int Id { get; set; }

  public int Source { get; set; }

  public int Destination { get; set; }

  public DateTime Date { get; set; }

  public string Status { get; set; }

  public string Network { get; set; }


  public NetworkLog(int id, int source, int destination, DateTime date, string status,
string network)

  {

    Id = id;

    Source = source;

    Destination = destination;

    Date = date;

    Status = status;

    Network = network;
  }

  public List<NetworkLog> ReadFile(string filePath)
  {
    List<NetworkLog> logs = new List<NetworkLog>();
    string[] lines = File.ReadAllLines(filePath);

    int id = 0, src = 0, dest = 0;
    DateTime date = DateTime.MaxValue;
    string status = "", netw = "";

    foreach (string line in lines)
    {
      if (line.StartsWith("Id"))
        id = int.Parse(line.Split(':')[1].Trim());
      else if (line.StartsWith("Source"))
        src = int.Parse(line.Split(':')[1].Trim());
      else if (line.StartsWith("Destination"))
        dest = int.Parse(line.Split(':')[1].Trim());
      else if (line.StartsWith("Date"))
      {
        string dateStr = line.Substring(line.IndexOf(':') + 1).Trim();
        date = DateTime.ParseExact(dateStr, "M/d/yyyy h:mm:ss tt",
System.Globalization.CultureInfo.InvariantCulture);
      }
      else if (line.StartsWith("Status"))
        status = line.Split(':')[1].Trim();
      else if (line.StartsWith("Network"))
```

```
          {
              netw = line.Split(':')[1].Trim();
              if (id != 0 && src != 0 && dest != 0 && date != DateTime.MaxValue &&
!string.IsNullOrEmpty(status) && !string.IsNullOrEmpty(netw))
              {
                  logs.Add(new NetworkLog(id, src, dest, date, status, netw));
                  id = 0; src = 0; dest = 0; date = DateTime.MaxValue; status = ""; netw = "";
              }
          }
      }
      return logs;
    }

    public void DisplayLogs(List<NetworkLog> logs)
    {
      Console.WriteLine("Id\tSource\tDestination\tDate\t\t\tTime\t\tStatus\tNetwork");
      Console.WriteLine(new string('-', 90));
      foreach (var item in logs)
      {

Console.WriteLine($"{item.Id}\t{item.Source}\t{item.Destination}\t{item.Date.ToShortD
ateString()}\t{item.Date.ToShortTimeString()}\t{item.Status}\t{item.Network}");
      }
    }
}

public class Program
{
    public static void Main(string[] args)
```

```
  {
    NetworkLog log = new NetworkLog(0, 0, 0, DateTime.MaxValue, "", "");
    string filePath = "networklog.txt";
    List<NetworkLog> logs = log.ReadFile(filePath);

    Console.WriteLine("\nSearch Options:");
    Console.WriteLine("1. Id\n2. Source\n3. Destination\n4. Date\n5. Status\n6.
Network");
    Console.Write("Enter your choice (1-6): ");
    string choice = Console.ReadLine();

    List<NetworkLog> filteredLogs = new List<NetworkLog>();

    switch (choice)
    {
      case "1":
        Console.Write("Enter Id to search: ");
        int searchId = int.Parse(Console.ReadLine());
        filteredLogs = logs.Where(l => l.Id == searchId).ToList();
        break;

      case "2":
        Console.Write("Enter Source to search: ");
        int searchSource = int.Parse(Console.ReadLine());
        filteredLogs = logs.Where(l => l.Source == searchSource).ToList();
        break;

      case "3":
        Console.Write("Enter Destination to search: ");
        int searchDestination = int.Parse(Console.ReadLine());
        filteredLogs = logs.Where(l => l.Destination == searchDestination).ToList();
        break;

      case "4":
        Console.Write("Enter Date (MM/DD/YYYY) to search: ");
        string dateInput = Console.ReadLine();
        if (DateTime.TryParse(dateInput, out DateTime searchDate))
        {
          filteredLogs = logs.Where(l => l.Date.Date == searchDate.Date).ToList();
        }
        else
        {
          Console.WriteLine("Invalid date format.");
        }
        break;

      case "5":
        Console.Write("Enter Status to search: ");
        string searchStatus = Console.ReadLine().Trim();
        filteredLogs = logs.Where(l => l.Status.Equals(searchStatus,
StringComparison.OrdinalIgnoreCase)).ToList();
        break;

      case "6":
        Console.Write("Enter Network to search: ");
        string searchNetwork = Console.ReadLine().Trim();
        filteredLogs = logs.Where(l => l.Network.Equals(searchNetwork,
StringComparison.OrdinalIgnoreCase)).ToList();
        break;


      default:
        Console.WriteLine("Invalid choice.");
        break;
    }


    if (filteredLogs.Any())
    {
      Console.WriteLine("\nFiltered Results:");
      log.DisplayLogs(filteredLogs);
    }
    else
    {
      Console.WriteLine("\nNo records found for the given criteria.");
    }
}
}
```