REPORTE DE PRÁCTICA

| Nombre Completo | Machado Sanchez Javier |
| --- | --- |
| Número de la Unidad a Evaluar | 3 |
| Tema a Evaluar | Lista Enlazada Ordenada Alfabéticamente Nombres |
| Fecha de Entrega | 14 de Octubre del 2023 |

**Programa Fuente**

| using System;  namespace P28\_Lista\_Enlazada\_Nombres\_Ascendientes  {  internal class Program  {  //---------------------------------------------------------------------------------------------  //Declaración de la clase Nodo  public class Node  { //Campo para almacenar la edad  public string name;  //Campo para almacenar la referencia al siguiente nodo  public Node next;  }  //Fin de la clase Nodo  //---------------------------------------------------------------------------------------------  //---------------------------------------------------------------------------------------------  //Declaración de variables globales  public static Node startNode, endNode, tempNode, aNode, sNode;  public static int length\_list = 0;  //Fin variables globales  //---------------------------------------------------------------------------------------------  //---------------------------------------------------------------------------------------------  //Declaración de método para insertar  public static void AddElement(string element)  {  tempNode = new Node();  tempNode.name = element;  if (startNode == null)  {  // Si la lista está vacía, crea un nuevo nodo y establece startNode y endNode en él.  startNode = tempNode;  endNode = tempNode;  tempNode.next = null;  length\_list++;  }  else if (element.CompareTo(startNode.name) < 0)  {  // Si el elemento es menor que el primer nodo, colócalo al principio.  tempNode.next = startNode;  startNode = tempNode;  length\_list++;  }  else  {  Node currentNode = startNode;  Node previousNode = null;  while (currentNode != null && element.CompareTo(currentNode.name) >= 0)  {  previousNode = currentNode;  currentNode = currentNode.next;  }  if (currentNode == null)  {  // Si el elemento es mayor que todos los nodos actuales, colócalo al final.  endNode.next = tempNode;  endNode = tempNode;  tempNode.next = null;  length\_list++;  }  else  {  // Inserta el elemento entre previousNode y currentNode.  previousNode.next = tempNode;  tempNode.next = currentNode;  length\_list++;  }  }  }  //Fin método para insertar  //---------------------------------------------------------------------------------------------  //---------------------------------------------------------------------------------------------  //Declaración de método para eliminar  public static bool DeleteElement(string element)  {  if (startNode == null)  {  // La lista está vacía, no se puede eliminar nada.  return false;  }  if (element == startNode.name)  {  if (startNode == endNode)  {  // Si solo hay un elemento en la lista.  startNode = null;  endNode = null;  }  else  {  // Actualiza las referencias cuando se elimina el primer elemento.  tempNode = startNode;  startNode = startNode.next;  endNode.next = startNode;  tempNode = null;  }  Program.length\_list--;  return true;  }  aNode = startNode;  sNode = startNode.next;  while (sNode != startNode)  {  if (element == sNode.name)  {  aNode.next = sNode.next;  sNode = null;  Program.length\_list--;  return true;  }  aNode = sNode;  sNode = sNode.next;  }  // Si el elemento no se encontró en toda la lista.  return false;  }  //Fin método para eliminar  //---------------------------------------------------------------------------------------------  //---------------------------------------------------------------------------------------------  //Declaración de método imprimir la lista  public static void PrintElements()  {  if (Program.length\_list > 0)  {  Console.WriteLine(Program.length\_list == 1  ? "\n\n\tEste es el nombre de la lista:"  : $"\n\n\tEstos son los {Program.length\_list} nombres de la lista:");  Node currentNode = startNode;  Console.Write("\n\t");  for (int i = 0; i < Program.length\_list; i++)  {  Console.Write($"[ {currentNode.name} ] ");  currentNode = currentNode.next;  if (currentNode == startNode)  {  break; // Detener el bucle cuando volvemos al primer elemento.  }  }  }  else  {  Console.WriteLine("\n\n\tLa lista se encuentra vacía.");  }  }  //Fin método para imprimir la lista  //---------------------------------------------------------------------------------------------  //---------------------------------------------------------------------------------------------  //Inicio Main  static void Main(string[] args)  {  Console.Title = "P28 Lista Enlazada con nombres ordenados por alfabéticamente";  //Declaración de variables locales  string name; char opcion\_menu; bool answer\_bool = false;  //Despliegue de menú  do  {  Console.Write("\n\t\t>> MENÚ DE OPCIONES <<" +  "\n\n\t1.- Insertar Elemento." +  "\n\t2.- Eliminar Elemento." +  "\n\t3.- Mostrar Elementos." +  "\n\t4.- Salir del Programa." +  "\n\n\tIngrese el número de la opción deseada: ");  //Condición para controlar excepción por formato incorrecto  if (char.TryParse(Console.ReadLine(), out opcion\_menu))  {  Console.Write("\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  //Switch case para controlar las opciones del menú  switch (opcion\_menu)  {  case '1':  {  do  {  Console.Write("\n\t\t>> INSERTANDO ELEMENTO <<" +  "\n\n\tPor favor, ingrese el nombre que desea insertar: ");  name = Console.ReadLine();  //Llamada a método para insertar  AddElement(name);  Console.Write($"\n\n\tSe ha insertado el nombre {name} en la lista.");  PrintElements(); //Llamada a método para imprimir  do  { //Ciclo do while para controlar las repuestas del usuario  Console.Write("\n\n\t¿Desea insertar otro nombre [S = Sí] [N = No] ?" +  "\n\tIngrese la letra de su respuesta: ");  //Llamada a DoAgain para controlar las respuestas  (answer\_bool, opcion\_menu) = DoAgain(Console.ReadLine(), answer\_bool);  } while (answer\_bool == false);  } while (opcion\_menu != 'N');  }  break;  case '2':  {  do  { //Condición para validar que la lista contenga elementos  if (length\_list > 0)  {  Console.Write("\n\t\t>> ELIMINANDO ELEMENTO <<");  PrintElements();  Console.Write("\n\n\tIngrese el nombre que desea eliminar: ");  name = Console.ReadLine();  if (DeleteElement(name))  {  Console.Write("\n\t\t>> ELIMINANDO ELEMENTO <<");  Console.Write( $"\n\n\n\tSe ha eliminado el nombre {name} correctamente.");  }  else  {  Console.Write("\n\n\t\t>> NO HAY NADA QUE ELIMINAR <<" +  $"\n\n\tNo se ha encontrado el nombre {name}.");  Console.Write("\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  PrintElements();  do  {  Console.Write("\n\n\t¿Desea eliminar otro nombre [S = Sí] [N = No] ?" +  "\n\tIngrese la letra de su respuesta: ");  //Llamada a DoAgain para controlar las respuestas  (answer\_bool, opcion\_menu) = DoAgain(Console.ReadLine(), answer\_bool);  } while (answer\_bool == false);  }  else  {  Console.Write("\n\t\t>> NO HAY NADA QUE ELIMINAR <<");  Console.WriteLine("\n\n\tLa lista se encuentra vacía.");  Console.Write("\n\tPresione la tecla <INTRO> para regresar al menú...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  opcion\_menu = 'N';  }  } while (opcion\_menu != 'N');  }  break;  case '3':  {  if (length\_list > 0)  {  Console.Write("\n\t\t>> MOSTRANDO ELEMENTOS <<");  //Llamada a método para imprimir  PrintElements();  Console.Write("\n\n\tPresione la tecla <INTRO> para regresar al menú...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  else  {  Console.Write("\n\t\t>> NO HAY NADA QUE MOSTRAR <<");  //Llamada a método para imprimir  PrintElements();  Console.Write("\n\n\tPresione la tecla <INTRO> para regresar al menú...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  opcion\_menu = 'N';  }  }  break;  case '4':  {  Console.Write("\n\t\t:. SALIDA :." +  "\n\n\tGracias por utilizar nuestro programa" +  "\n\n\tPresione la tecla <Esc> para salir...");  while (Console.ReadKey().Key != ConsoleKey.Escape) ;  }  break;  default:  {  Console.Write("\n\t\t.: OPCIÓN INVALIDA :." +  "\n\n\tPor favor ingrese una opción existente" +  "\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  break;  }  }  else  {  Console.Write("\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  Console.Write("\n\t\t>> ALGO HA SALIDO MAL <<" +  "\n\n\tPor favor ingrese solamente un número." +  "\n\n\tPresione la tecla <INTRO> para regresar al menú...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  } while (opcion\_menu != '4');  }  //Fin Main  //---------------------------------------------------------------------------------------------  //Inicio método para preguntar hacer de nuevo  /\*Definición de método DoAgain(Hacer de nuevo) para hacer todo el procesamiento  \* de las respuestas leídas.  \* Answer bool es true cuando la respuesta es 'S' o 'N', por lo que se puede salir del  \* do while donde se hace la pregunta. En caso de ser cualquier otra respuesta se avisa  \* al usuario que la opción no existe y se vuelve a imprimir la pregunta  \*/  public static (bool, char) DoAgain(string read\_line, bool answer\_bool)  {  char opcion\_menu;  if (char.TryParse(read\_line, out opcion\_menu))  {  opcion\_menu = char.ToUpper(opcion\_menu);  if (opcion\_menu == 'S')  {  Console.Write("\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  return (answer\_bool = true, opcion\_menu);  }  else if (opcion\_menu == 'N')  {  Console.Write("\n\tPresione la tecla <INTRO> para regresar al menú...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  return (answer\_bool = true, opcion\_menu);  }  else  {  Console.Write("\n\n\tPor favor, ingrese una respuesta válida" +  "\n\tPresione la tecla <INTRO> para intentarlo de nuevo...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  return (answer\_bool = false, opcion\_menu);  }  }  else  {  Console.Write("\n\n\tPor favor, ingrese una respuesta válida" +  "\n\tPresione la tecla <INTRO> para intentarlo de nuevo...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  return (answer\_bool = false, opcion\_menu);  }  //Fin método para preguntar hacer de nuevo  //---------------------------------------------------------------------------------------------  }  } |
| --- |

**Ventana de Entrada de datos**

|  |
| --- |

**Ventana de Resultados**

|  |
| --- |

**EXPLICA BREVEMENTE QUÉ TE PARECIÓ EL PROBLEMA, QUE SE TE DIFICULTO Y COMO LO SOLUCIONASTE.**

| Gran parte del programa lo copie de el p27 y solamente lo adapté para ordenar alfabéticamente apoyándome también del vídeo de la profesora sobre colas circulares. Durante el hackathon estuvimos parte de mí equipo y yo tratando de realizar este programa también(nuestros brakes de programación eran seguir programando pero en algo más fácil como c# y no en la complejidad de la blockchain de Solana JAJA) y ahí pudimos lograr que se insertaran correctamente todos los nombres y se mostraran también, sin embargo, el programa no nos permitía eliminar los nombres del centro, solamente el primero y último.  Con ayuda de chatgpt pudimos hacer que se eliminaran todos sin importar el lugar, a excepción del primer elemento, el cual al eliminarlo se iniciaba un bucle infinito donde se mostraba una y otra vez la lista.  Esto lo solucione al pedirle a chatgpt que volviera a construir el bucle para mostrar los datos, pero que usara for en lugar del while, haciendo así que el bucle infinito terminara y el programa funcione al 100.  También una parte del código que era así:  if(Program.length\_list == 1) Console.WriteLine(“\n\n\tEste es el nombre de la lista:”);  else Cosole.WriteLine(“\n\n\tEstos son los {Program.length\_list} nombres de la lista:”);  Logre simplificarlo de la siguiente manera:  Console.WriteLine(Program.length\_list == 1 ? "\n\n\tEste es el nombre de la lista:" : $"\n\n\tEstos son los {Program.length\_list} nombres de la lista:");  utilizando el “Operador Condicional Ternario” el cual su sintaxis es  condición ? resultado true : resultado false  Esto lo aprendí debido a que en el hackathon lo utilizamos mucho al trabajar con las llamadas al wallet de solana o al estar importando información de API’s, así que me pareció muy genial poder implementar algo aprendido en JavaScript durante este evento a mí código de C#. |
| --- |