REPORTE DE PRÁCTICA

| Nombre Completo | Machado Sanchez Javier |
| --- | --- |
| Número de la Unidad a Evaluar | 5 |
| Tema a Evaluar | Ordenamiento por Shell Ascendente |
| Fecha de Entrega | 01 de Noviembre del 2023 |

**Programa Fuente**

| using System;  namespace P34\_Ordenamiento\_por\_Shell\_Ascendente  {  internal class Program  {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //Declaración de método para ordenar  public static short[] OrdenamientoShell(short[] Calificaciones)  {  //Inicio variables locales  bool flag = false;  int salto = 0;  short t;  //Fin variables locales  //Encontrando la mitad del arreglo  salto = Calificaciones.Length / 2;  //Ciclos anidados para hacer el ordenamiento  while(salto > 0)  {  flag = true;  while(flag != false)  {  flag = false;  for (int i = 1; i <= Calificaciones.Length - salto; i++)  {  if (Calificaciones[i - 1] > Calificaciones[i - 1 + salto])  {  t = Calificaciones[i - 1 + salto];  Calificaciones[i - 1 + salto] = Calificaciones[i - 1];  Calificaciones[i - 1] = t;  flag = true;  }  }  }  salto = salto / 2;  }  return Calificaciones;  }  //Fin declaración de método para ordenar  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //Inicio declaración método para mostrar  public static void Mostrar(short[] Calificaciones)  {  for (int i = 1; i <= Calificaciones.Length; i++)  {  Console.Write($"[{Calificaciones[i - 1]}] ");  }  }  //Fin declaración de método mostrar  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  static void Main(string[] args)  {  Console.Title = "P34 Shell Sort Ascendente";  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //Declaración de variables locales  short[] Calificaciones = new short[20];  short Calificacion; char opcion;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  do //Do while para controlar menú  {  //Despliegue de menú  Console.Write("\n\t\t.: MENÚ DE OPCIONES :." +  "\n\n\t[ 1 ] - Capturar Calificaciones" +  "\n\t[ 2 ] - Ordenar Calificaciones Ascendentemente" +  "\n\t[ 3 ] - Salir del Programa" +  "\n\n\tIngrese el número de la opción deseada: ");  if (char.TryParse(Console.ReadLine(), out opcion) != false) //Try catch para controlar errores de formato  {  Console.Write($"\n\n\tLa opción {opcion} se ha capturado correctamente" +  $"\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  //Switch case para controlar opciones  switch (opcion)  {  case '1':  {  Console.WriteLine("\n\t.: CAPTURA DE DATOS :.");  for (int i = 1; i <= Calificaciones.Length; i++) //Ciclo para capturar todas las calificaciones  {  Console.Write($"\n\t{i}.- Ingrese una califiación: ");  if (short.TryParse(Console.ReadLine(), out Calificacion) == true && Calificacion <= 100 && Calificacion >= 0)  {  Console.WriteLine($"\n\tSe ha capturado la calificación {Calificacion} correctamente.");  Calificaciones[i - 1] = Calificacion;  }  else  {  Console.WriteLine($"\n\tLa calificación debe ser un valor entero entre 1 y 100");  i--;  }  }  Console.Write("\n\tCaptura de datos terminada, presione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  break;  case '2':  {  Console.WriteLine("\n\t\t.: ORDENANDO DATOS ASCENDENTEMENTE :." +  "\n\n\t| Este es el arreglo original sin ordenar |\n\t");  Mostrar(Calificaciones); //Llamada a método para mostrar  Console.WriteLine("\n\n\t| Este es el arreglo ordenado ascendentemente |\n\t");  Calificaciones = OrdenamientoShell(Calificaciones);//llamada a método para ordenar  Mostrar(Calificaciones);  Console.Write("\n\tOrdenamiento terminado, presione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  break;  case '3':  {  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\tLa opción no se ha podido capturar, asegúrese de escribirla correctamente" +  "\n\n\tPresione la tecla <INTRO> para continuar...");  while (Console.ReadKey().Key != ConsoleKey.Enter) ; Console.Clear();  }  } while (opcion != '3');  }  }  } |
| --- |

**Ventana de Entrada de datos**

|  |
| --- |

**Ventana de Resultados**

|  |
| --- |

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

| No conocía este tipo de ordenamiento ni lo había implementado nunca. La explicación en clase y los vídeos de la profesora fueron de gran ayuda para poder modificar el código del P32 y adaptarlo a este nuevo ordenamiento. |
| --- |