HOMEWORK: 22/09/2023

NAME: LÊ HỮU HOÀN

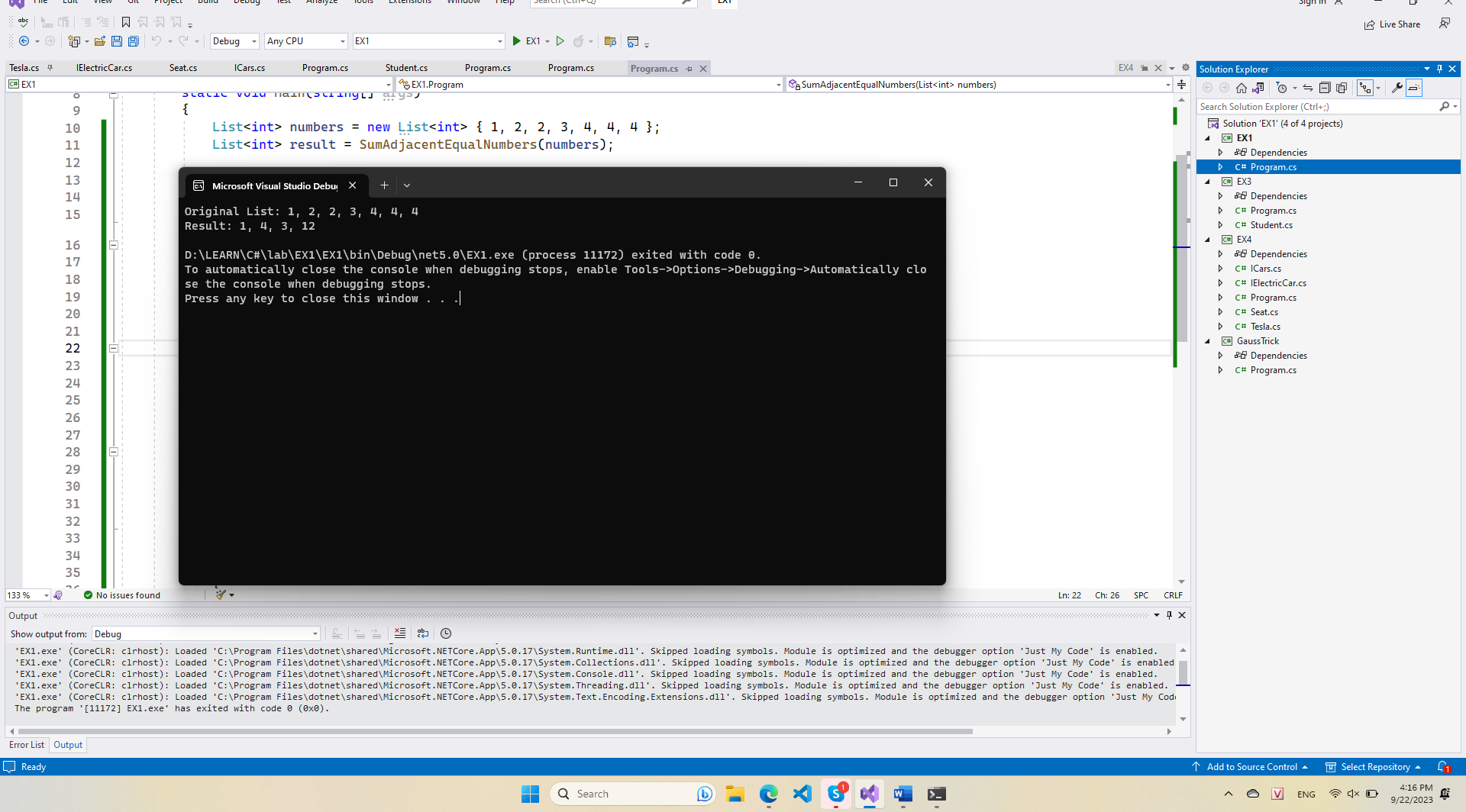
CLASS: Lập trình C# (3)

Exercise 1: Sum Adjacent Equal Numbers

Write a program to sum all adjacent equal numbers in a list of decimal numbers,

starting from left to right.

CONSOLE



CODE

|  |
| --- |
| class Program  {  static void Main(string[] args)  {  List<int> numbers = new List<int> { 1, 2, 2, 3, 4, 4, 4 };  List<int> result = SumAdjacentEqualNumbers(numbers);  Console.WriteLine("Original List: " + string.Join(", ", numbers));  Console.WriteLine("Result: " + string.Join(", ", result));  }  static List<int> SumAdjacentEqualNumbers(List<int> numbers)  {  List<int> result = new List<int>();  int i = 0;  int n = numbers.Count;  while (i < n)  {  int currentNumber = numbers[i];  int sum = currentNumber;  // Check if the next number is equal to the current number  while (i + 1 < n && numbers[i + 1] == currentNumber)  {  sum += numbers[i + 1];  i++;  }  result.Add(sum);  i++;  }  return result;  }  } |

Exercise 2: Gauss’’ Trick

Write a program that sum all numbers in a list in the following order:

first + last, first + 1 + last - 1, first + 2 + last - 2, … first + n, last – n

CONSOLE

A computer screen with a black screen

Description automatically generated

CODE

|  |
| --- |
| class Program  {  static void Main(string[] args)  {  List<int> numbers = new List<int> { 1, 2, 3, 4, 5 };  int[] result = GaussTrick(numbers);  Console.WriteLine("Original List: " + string.Join(", ", numbers));  Console.WriteLine("Result: " + string.Join(", ", result));  }  static int[] GaussTrick(List<int> numbers)  {  int n = numbers.Count;  int[] result = new int[n / 2 + n % 2];  for (int i = 0; i < n / 2; i++)  {  result[i] = numbers[i] + numbers[n - i - 1];  }  if (n % 2 != 0)  {  result[n / 2] = numbers[n / 2];  }  return result;  }  } |

Exercise 3

- Define a class Student, which holds the following information: ID, Name and Age.

- Create a list of students

- Sort the list by Name and print it.

CONSOLE

A computer screen shot of a black screen

Description automatically generated

CODE

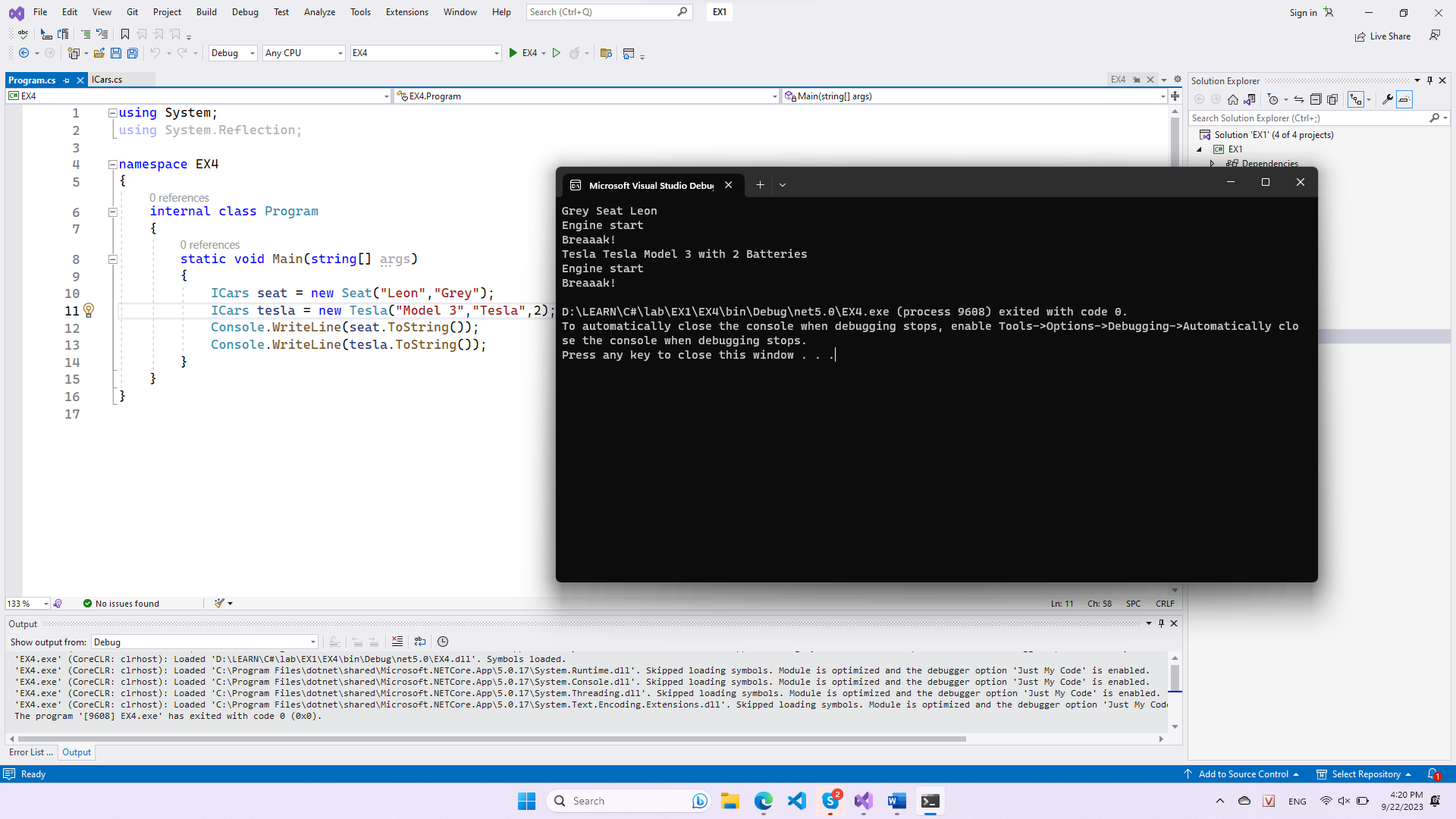
|  |
| --- |
| class Student  {  public int ID { get; set; }  public string Name { get; set; }  public int Age { get; set; }  } |

|  |
| --- |
| class Program  {  static void Main(string[] args)  {  List<Student> students = new List<Student>  {  new Student { ID = 1, Name = "John", Age = 20 },  new Student { ID = 2, Name = "Alice", Age = 22 },  new Student { ID = 3, Name = "Bob", Age = 21 },  new Student { ID = 4, Name = "Emma", Age = 19 }  };  // Sort the list by Name  students.Sort((student1, student2) => student1.Name.CompareTo(student2.Name));  // Print the sorted list  foreach (Student student in students)  {  Console.WriteLine($"ID: {student.ID}, Name: {student.Name}, Age: {student.Age}");  }  }  } |

Exercise 4: Cars

NOTE: You need a public StartUp class with the namespace Cars.

Build a hierarchy of interfaces and classes:



CODE

|  |
| --- |
| class Program  {  static void Main(string[] args)  {  ICars seat = new Seat("Leon","Grey");  ICars tesla = new Tesla("Model 3","Tesla",2);  Console.WriteLine(seat.ToString());  Console.WriteLine(tesla.ToString());  }  } |

|  |
| --- |
| interface ICars  {  string Model { get; set; }  string Color { get; set; }  string Start();  string Stop();  string ToString();  } |

|  |
| --- |
| interface IElectricCar  {  int Battery { get; set; }  } |

|  |
| --- |
| class Seat : ICars  {  public string Model { get; set; }  public string Color { get; set ; }  public Seat(string model, string color)  {  Model = model;  Color = color;  }  public string Start()  {  return "Engine start";  }  public string Stop()  {  return "Breaaak!";  }  public new string ToString()  {  return $"{Color} Seat {Model} \n{Start()}\n{Stop()}";  }  } |

|  |
| --- |
| class Tesla : ICars, IElectricCar  {  public string Model { get; set; }  public string Color { get; set; }  public int Battery { get; set; }  public Tesla(string model, string color, int battery)  {  Model = model;  Color = color;  Battery = battery;  }  public string Start()  {  return "Engine start";  }  public string Stop()  {  return "Breaaak!";  }  public new string ToString(){  return $"{Color} Tesla {Model} with {Battery} Batteries \n{Start()}\n{Stop()}";  }  } |