Day 20 Assignment

By

B.P.N.V.S.Sudheer

18-02-22

|  |
| --- |
| 1.What are delegates in C# Write the points dicussed about delegates in the class Write C# code to illustrate the usage of delegates. |
| Delegates allow methods to be passed as parameters. Delegates can be used to define callback methods. Delegates can be chained together; for example, multiple methods can be called on a single event. Methods don't have to match the delegate type exactly |
| * A Delegate is like a function pointer |
| * Using delegates we can call or point to one or more methods |
| * When declaring a delegate return type and parameters must match with the methods you want to point using the delegate |
| * Benefits of delegate is that using single call from delegate all your methods |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace day20project1  {  public delegate void MyCaller(int a, int b);  internal class Program  {  public static void Add(int a, int b)  {  Console.WriteLine(a + b);  }    public static void Mul(int a, int b)  {  Console.WriteLine(a \* b);  }    public static void Div(int a, int b)  {  Console.WriteLine(a / b);  }  static void Main(string[] args)  {  MyCaller mc = new MyCaller(Add);  mc += Mul;  mc += Div;    //10,20  mc(10, 20);    //20,30  mc(20, 30);    //80,90  mc(80, 90);    Console.ReadLine();  }  }  } |

|  |
| --- |
| Output: |
|  |
| 2. What are nullable types in C#WACP to illustrate nullable types Write some properties of nullable types (like HasValue) |
| A nullable type is a special version of the value type that is represented by a structure. In addition to the values defined by the underlying type, a nullable type can also store the value null |
| Code : |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace day20project2  {  internal class Program  {  static void Main(string[] args)  {  byte? age = null;  Console.WriteLine(age);      Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
|  |
| Code : |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace day20project2  {  internal class Program  {  static void Main(string[] args)  {  byte? input = null;    if (input.HasValue)  {  Console.WriteLine(input \* input);  }  else  {  Console.WriteLine("No Value");  }  Console.ReadLine();      Console.ReadLine();  }  }  } |
| Output: |
|  |
| 3.out, ref - parameters please research on these two types of parameters write a C# program to illustrate the same. |
| Ref Parameter: |
| ref is used to state that the parameter passed may be modified by the method. in is used to state that the parameter passed cannot be modified by the method |

|  |
| --- |
| Code : |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace day20project4  {  internal class Program  {  static void Main(string[] args)  {  int myNum = 10;        Console.WriteLine(myNum);    Console.ReadLine();  }  public static void Number(int num)  {  num = 100;  }  }  } |
| Output: |
|  |
| Out parameter: |
| out is used to state that the parameter passed must be modified by the method |
| Code : |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace day20project3  {  internal class Program  {  static void Main(string[] args)  {  int x;  Multiplication(out x);  Console.WriteLine(x);  Console.ReadLine();  }  public static void Multiplication(out int a)  {  a = 10;  a \*= a;  }  }  } |

|  |
| --- |
| Output: |
|  |
| 4.Research and understand scope of variables in C# |
| In general, the scope is defined as the extent up to which something can be worked with. In programming also the scope of a variable is defined as the extent of the program code within which the variable can be accessed or declared or worked with |
|  |
|  |