|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | DAY 20 ASSIGNMENTS  DATE :18/02/2022  DAY :FRIDAY  BY  M. SAI HARI CHANDANA | | 1.Research and understand scope of variables in C# ? | | * The part of the program where a particular variable is accessible is termed as the scope of that variable * A variable can be defined in aclass method, loop, * A variable can be determined at compile time and independent of the function call stack * Scope of variables has three types   A.Class level scope  B. Method level scope  C. Block level scope | | 1. CLASS LEVEL SCOPE : | | * Declaring the variables in aclass but outside any method can be directly accessed anywhere in the class * These variables also termed as the fields or class members * Access modifier of class level variables doesnot affect their scope within a class | | 1. METHOD LEVEL SCOPE : | | * Variables that are declared inside a method have method level scope these are not accessible outside the method * These variables are termed as the local variables * These variables don,t exist after methods execution is over | | 1. BLOCK LEVEL SCOPE : | | * These variables are generally declared inside the for ,while statement etc. * These variables are also termed as the loop variables or statements variable as they have limited their scope up to the body of the statement in which it * A variable which is declared inside a loop body will not be visible to the outside of loop body |      |  | | --- | | 2.what are delegates in c# ?  Write the points discussed about delegates in the class  Write c# code to illustrate the usage of delegates ? | | * DELEGATES IN C#   A delegate is a type that represents reference to methods with a particular parameter list and return type . | | PONITS OF DELEGATES : | | * A delegates is like a function pointer * Using delegates we can call (or ) point to one or more methods * When declaring a delegate return and parameters must match with the methods you want to point using the delegate * Benefit of delegate is that using single call from delegate , all your methods pointing to delegate will be called * It has 2 types * Single cast ,multi cast | | CODE : | | sing System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace ConsoleApp1  {  public delegate void MyCaller(int a, int b);  internal class Program  {  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;  //12,6  mc(12, 6);  //20,10  mc(20, 10);  //16,8  mc(16, 8);  Console.ReadLine();  }  }  }  } | | Output : |  |  | | --- | | 3.what are nullable types in c#  WACP to illustrate nullable types  Write some properties of nullable types (like has value ) | | * Nullable types allows you to assign a null value to variable nullable type can work with value type not with reference type | | CODE : | | using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace ConsoleApp2  {  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();  }  }  } | | OUTPUT : | | 4.out ,ref parameters  Please research on these two parameters  WACP to illustrate the same | | OUT PARAMETERS :   * .OUT parameter is used when a method return multiple values when aparameter passes with the out keyword /parameter in the method then that method works with the same variable * That is passed in the method call if value changes the method parameter value also changes | | REFERRENCE PARAMETERS : | | A reference parameter is a reference to a memory location of variable when you pass parameters by refrence unlike value parameters a new storage location is not created for these parameters the same memory location as the actual parameters that are supplied to the method | | CODE : | | using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace ConsoleApp3  {  internal class Program  {  public static void Update(out int a)  {  a = 10;  }  public static void Change (ref int d)  {  d = 11;  }  static void Main(string[]args)  {  int b;  int c = 5;  Program p1=new Program();  Update(out b);  Change(ref c);  Console.WriteLine("Updated value is :{0},52");  Console.WriteLine("changed value is :{0},61");  Console.ReadLine();  }      }          } | | OUTOUT : | |