

Definition:

should have a Point class which should have two public int variables x and y.
initialize it with 100 and 200
create another class Program
create a method static MyMethod which accept 2 parameters Point type p1 and p2
call from main method

p1.x +p2.x

p1.y +p2.y in method

Lab Ex

```
using System;

namespace DotNet
{
    class Point
    {
        public int x;
        public int y;
    }

    class Program
    {
        static void MyMethod(Point p1,Point p2){
            Console.WriteLine($"P1.X+p2.X = {p1.x+p2.x}");
            Console.WriteLine($"P1.Y+p2.Y ={p1.y+p2.y}");
        }

        static void Main(string[] args)
        {
            Point p1 = new Point();
            Point p2 = new Point();
            Console.Write("Enter Value Of P1 X :");
            p1.x = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P1 Y :");
            p1.y = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P2 X :");
            p2.x = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P2 Y :");
            p2.y = Convert.ToInt32(Console.ReadLine());

            MyMethod(p1,p2);
        }
    }
}
```

Lab Ex Using ref Keyword

```
using System;

namespace DotNet
{
    class Point
    {
        public int x;
        public int y;
    }

    class Program
    {
        static void MyMethod(ref Point p1, ref Point p2)
        {
            Console.WriteLine($"P1.X+p2.X = {p1.x+p2.x}");
            Console.WriteLine($"P1.Y+p2.Y ={p1.y+p2.y}");
        }

        static void Main(string[] args)
        {
            Point p1 = new Point();
            Point p2 = new Point();
            Console.Write("Enter Value Of P1 X :");
            p1.x = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P1 Y :");
            p1.y = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P2 X :");
            p2.x = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter Value Of P2 Y :");
            p2.y = Convert.ToInt32(Console.ReadLine());

            MyMethod(ref p1, ref p2);
        }
    }
}
```

Lab Ex Using out Keyword

```
using System;

namespace DotNet
{
    class Point
    {
        public int x;
        public int y;
    }

    class Program
    {
        static void MyMethod(out Point p1,out Point p2){
            p1 = new Point();
            p2 = new Point();
            Console.WriteLine("Enter Value Of P1 X :");
            p1.x = Convert.ToInt32(Console.ReadLine());

            Console.WriteLine("Enter Value Of P1 Y :");
            p1.y = Convert.ToInt32(Console.ReadLine());

            Console.WriteLine("Enter Value Of P2 X :");
            p2.x = Convert.ToInt32(Console.ReadLine());

            Console.WriteLine("Enter Value Of P2 Y :");
            p2.y = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine($"P1.X+p2.X = {p1.x+p2.x}");
            Console.WriteLine($"P1.Y+p2.Y ={p1.y+p2.y}");
        }

        static void Main(string[] args)
        {
            Point p1 =null;
            Point p2 = null;

            MyMethod(out p1,out p2);
        }
    }
}
```

Parameter Array

```
using System;

namespace DotNet
{

    class Program
    {
        static void Sum(params int[] values) {
            int total=0;
            for(int i=0;i<values.Length;i++) {
                total +=values[i];
            }
            Console.WriteLine(${total});
        }

        static void PrintName(params string[] names) {
            for(int i=0;i<names.Length;i++) {
                Console.WriteLine(${names[i]});
            }
        }

        static void Main(string[] args)
        {
            Sum(10,20,30);
            PrintName("Ankit", "Rudani");
        }
    }
}
```

Parameter Array find Square and Cube

```
using System;

namespace DotNet
{
    class Program
    {
        static void SquareCube(params int[] nos)
        {
            Console.WriteLine("-----");
            for(int i=0;i<nos.Length;i++)
            {
                Console.WriteLine($"{nos[i]}^2 = {nos[i]*nos[i]} ,");
                nos[i]^3 = {nos[i]*nos[i]*nos[i]};
            }
        }

        static void Main(string[] args)
        {
            SquareCube(1,2,3,4);
            SquareCube(5,6,7,8);
        }
    }
}
```

Value Parameter	Reference Parameter	Output Parameter
No keywords are used while passing and receiving values to the function.	While passing values and receiving value to the function "ref" keyword is used.	While passing values and receiving value to the function "out" keyword is used.
Actual parameter is copied to the formal parameter and new memory location is created for formal parameters.	Actual parameter works as alias to the formal parameter, and hence new memory location is not created for formal parameters	Actual parameter works as alias to the formal parameter, and hence new memory location is not created for formal parameters
Changes in formal parameters inside method does not affect actual parameter in case of value type.	Changes in formal parameters inside method affects actual parameter in case of value type & reference type	Changes in formal parameters inside method affects actual parameter in case of value type & reference type
Inside method it is not necessary to assign formal parameters.	Inside method it is not necessary to assign formal parameters.	Inside method it is mandatory to assign formal parameters.
MyMethod(ref_type, value_type);	MyMethod(ref/ref_type, ref value_type);	MyMethod(out ref_type, out value_type);

Named Parameters

The screenshot shows a Visual Studio Code window with the title "10.9.282 (DESKTOP-2ONU3B3) - VNC Viewer". The code editor displays a C# file named Program.cs with the following content:

```
4 class Program
5 {
6     static void PrintFullName(String fnm, String mn, String lnm)
7     {
8         Console.WriteLine($"{fnm} " + mn + " " + lnm);
9     }
10    static void Main(string[] args)
11    {
12        PrintFullName("Sumit", "Arvindbhai", "Arora");
13        PrintFullName(mn: "Arvindbhai", lnm: "Arora", fnm: "Sumit");
14    }
15}
16}
17}
18}
```

The terminal below shows the output of running the application with the command "dotnet run".

```
PS D:\parameters> dotnet run
Sumit Arvindbhai Arora
Sumit Arvindbhai Arora
PS D:\parameters> [ ]
```

Optional Parameters

The screenshot shows a Visual Studio Code window with the title "10.9.282 (DESKTOP-2ONU3B3) - VNC Viewer". The code editor displays a C# file named Program.cs with the following content:

```
1 using System;
2
3 namespace parameters
4 {
5     class Program
6     {
7         static void MyMethod(String name, String university = "Atmiya University")
8         {
9             Console.WriteLine($"Name - {name} , University - {university}");
10        }
11        static void Main(string[] args)
12        {
13            MyMethod("ABC", "Atmiya University");
14            MyMethod("DEF", "Saurashtra University");
15            MyMethod("GHI", "Atmiya University");
16            MyMethod("JKL");
17            MyMethod("MNO", "Gujarat Technological University");
18        }
19    }
20}
```

The terminal below shows the output of running the application with the command "dotnet run".

```
PS D:\parameters> dotnet run
Name = ABC , University = Atmiya University
Name = DEF , University = Saurashtra University
Name = GHI , University = Atmiya University
Name = JKL , University = Atmiya University
Name = MNO , University = Gujarat Technological University
PS D:\parameters> [ ]
```

Definition :

Write a C# Program to Create a class Employee which has two public variables name, designation and salary (int)

Program Method DispalyDetails(Employee e1)
if Designation is not pass then by default assign "Jr.SW"

```
using System;

namespace DotNet
{
    class Employee
    {
        public string name, designation="Jr. Software Developer";
        public int salary;
    }
    class Program
    {
        static void DisplayDetail(Employee e)
        {
            Console.WriteLine($"Name = {e.name}, Designation =
{e.designation}, Salary = {e.salary}");
        }
        static void Main(string[] args)
        {
            Employee e1 = new Employee();
            e1.name = "Ankit";
            e1.designation = "DBA";
            e1.salary = 1000;
            DisplayDetail(e1);

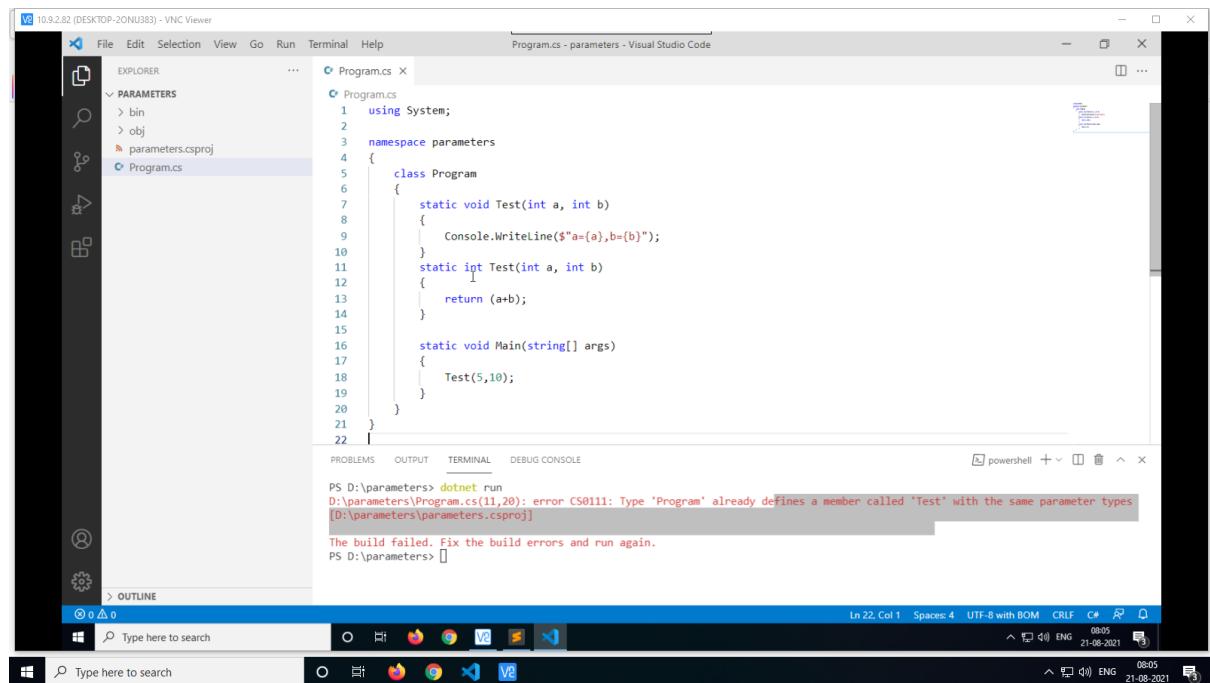
            Employee e2 = new Employee();
            e2.name = "Ankur";
            e2.salary = 1000;
            DisplayDetail(e2);
        }
    }
}
```

Method Overloading:

When a class have more than one method with same name is called method overloading, but they should have different signatures.

Method signatures includes:

1. Name of the Method
- 2.



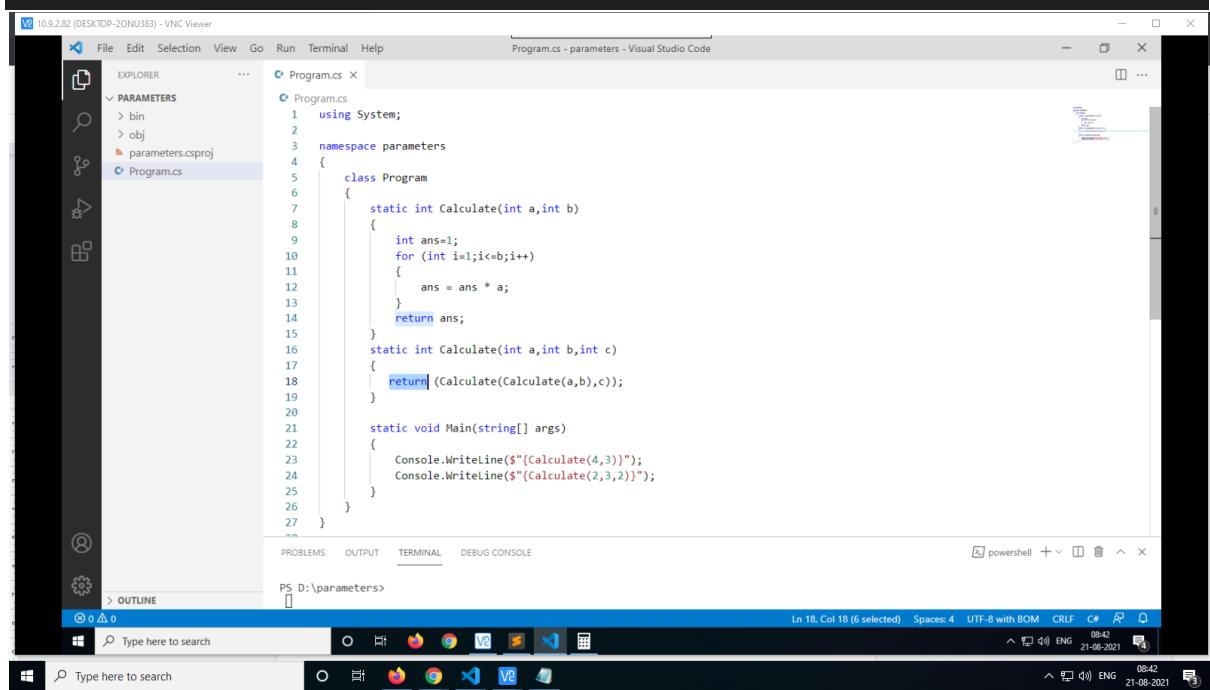
```
1  using System;
2
3  namespace parameters
4  {
5      class Program
6      {
7          static void Test(int a, int b)
8          {
9              Console.WriteLine($"a={a},b={b}");
10         }
11         static int Test(int a, int b)
12         {
13             return (a+b);
14         }
15
16         static void Main(string[] args)
17         {
18             Test(5,10);
19         }
20     }
21 }
```

Definition:

Write a C# program to overload a calculate method which is overloaded with no of parameters is to parameters are pass it will return a^b if three parameters are passed it will return $(a^b)^c$

```
using System;
namespace DotNet
{
    class Program
    {
        static int Calculate(int a,int b)
        {
            int ans = 1;
            for(int i=0;i<b;i++)
                ans *=a;
            return ans;
        }
    }
}
```

```
static int Calculate(int a,int b,int c){// Overloaded Method
    int ans = Calculate(a,b);
    return Calculate(ans,c);
}
static void Main(string[] args)
{
    Console.WriteLine(${Calculate(2,3)}");
    Console.WriteLine(${Calculate(2,3,4)}");
}
}
```



Static Constructor

Definition :

create a class Employee which is the static variable noe and non static members name and salary

create a parameterized constructor which takes name and salary as parameter to initialize instance members.

when new employee instance is created noe variable should be ++

class should have method to get name and salary and to print the name and salary and a method to print no of employees.

Answer

```
using System;

namespace DotNet
{
    class Employee{
        public static int noe=0;
        public string name;
        public int salary;

        public Employee(){
            name = "abc";
            salary = 12000;
            noe++;
        }

        public Employee(string nm,int sal){
            name = nm;
            salary = sal;
            noe++;
        }

        // static Employee(){
        //     noe++;
        //     Console.WriteLine("Hello");
        // }

        public void getDetails(){
            Console.WriteLine($"Name = {name}\nSalary = {salary}");
            Console.WriteLine("-----");
        }
    }
}
```

```
}

public static void getNOE(){
    Console.WriteLine($"No of Employee = {noe}");
}

class Program
{
    static void Main(string[] args)
    {
        Employee e1 = new Employee();
        e1.getDetails();

        Employee e2 = new Employee("Ankit",10000);
        e2.getDetails();
        Employee.getNOE();

        Employee e3 = new Employee("Ankur",15000);
        e3.getDetails();
        //Employee.getNOE();

        Employee e4 = new Employee("Kaushal",20000);
        e4.getDetails();
        Employee.getNOE();
    }
}
```

The screenshot shows a Visual Studio Code window with the title "Program.cs - parameters - Visual Studio Code". The code editor displays the following C# code:

```
1 //> static constructor is invoked
2
3 class Program
4 {
5     static void Main(string[] args)
6     {
7         MyClass m = new MyClass(5);
8     }
9 }
10 }
```

The code editor highlights line 19 with a red squiggle under "Console.WriteLine("static constructor is invoked");". A tooltip above the line says "static constructor must be parameterless". The status bar at the bottom shows the error message: "D:\parameters\Program.cs(19,16): error CS0132: 'MyClass.MyClass(int)': a static constructor must be parameterless [D:\parameters\parameters.csproj]".

A screenshot of Visual Studio Code running on a Windows desktop. The window title is "10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer". The code editor shows "Program.cs - parameters - Visual Studio Code". The code contains the following:

```
5 class MyClass
6 {
7     public static int a=10;
8     public int b = 20;
9
10    static MyClass() //instance constructor
11    {
12        Console.WriteLine($"a={a},b={b}");
13        a = a + 5;
14        b = b + 5;
15        Console.WriteLine($"a={a},b={b}");
16    }
17 }
18 class Program
19 {
```

The terminal output shows several errors:

```
D:\parameters\Program.cs(12,42): error CS0120: An object reference is required for the non-static field, method, or property 'MyClass.b' [D:\parameters\parameters.csproj]
D:\parameters\Program.cs(14,13): error CS0120: An object reference is required for the non-static field, method, or property 'MyClass.b' [D:\parameters\parameters.csproj]
D:\parameters\Program.cs(14,17): error CS0120: An object reference is required for the non-static field, method, or property 'MyClass.b' [D:\parameters\parameters.csproj]
D:\parameters\Program.cs(15,42): error CS0120: An object reference is required for the non-static field, method, or property 'MyClass.b' [D:\parameters\parameters.csproj]
```

The build failed. Fix the build errors and run again.

PS D:\parameters> █

Object Initializer

A screenshot of Visual Studio Code running on a Windows desktop. The window title is "10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer". The code editor shows "Program.cs - parameters - Visual Studio Code". The code contains the following:

```
1 using System;
2
3 namespace parameters
4 {
5     public class Point
6     {
7         public int X = 1;
8         public int Y = 2;
9     }
10    class Program
11    {
12        static void Main(string[] args)
13        {
14            Point p1 = new Point();
15            Point p2 = new Point {X=5, Y=6};
16
17            Console.WriteLine($"{p1.X},{p1.Y}");
18            Console.WriteLine($"{p2.X},{p2.Y}");
19        }
20    }
}
```

The terminal output shows the results of the execution:

```
PS D:\parameters> dotnet run
1,2
5,6
PS D:\parameters> █
```

Readonly

A screenshot of Visual Studio Code showing a C# file named Program.cs. The code defines a class Temp with a public readonly int x=10; and a class Program with a static void Main method that creates a Temp object and prints its value. The terminal shows the output of running the program.

```
1  using System;
2
3  namespace parameters
4  {
5      class Temp
6      {
7          public readonly int x=10;
8      }
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             Temp t = new Temp();
14             Console.WriteLine($"{t.x}");
15         }
16     }
17 }
18
19
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
10
PS D:\parameters>

A screenshot of Visual Studio Code showing the same C# program. The code has been modified to add a line t.x += 10; before the second Console.WriteLine statement. The terminal shows a build error: CS0191: A readonly field cannot be assigned to (except in a constructor or init-only setter). The build failed, and the user is instructed to fix the errors and run again.

```
1  using System;
2
3  namespace parameters
4  {
5      class Temp
6      {
7          public readonly int x=10;
8      }
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             Temp t = new Temp();
14             Console.WriteLine($"{t.x}");
15             t.x += 10;
16             Console.WriteLine($"{t.x}");
17         }
18     }
19 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
D:\parameters\Program.cs(15,13): error CS0191: A readonly field cannot be assigned to (except in a constructor or init-only setter) [D:\parameters\parameters.csproj]
The build failed. Fix the build errors and run again.
PS D:\parameters>

The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-2ONU3B3) - VNC Viewer
- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Editor:** Program.cs - parameters - Visual Studio Code. The code defines a class Temp with a static readonly field x=10, and a class Program with a Main method that prints "Temp.x".
- Terminal:** powershell. It shows the command "dotnet run" being run in the directory D:\parameters, which outputs "Temp.x".
- Sidebar:** EXPLORER, PARAMETERS (bin, obj), parameters.csproj, Program.cs.
- Bottom Status Bar:** Ln 13, Col 38, Spaces: 4, UTF-8 with BOM, CRLF, 0846, ENG, 24-08-2021. It also includes search bars for the editor and taskbar.

The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-2ONU3B3) - VNC Viewer
- Menu Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Code Editor:** Program.cs - parameters - Visual Studio Code. The code defines a class Temp with a static readonly field x and a constructor setting it to 10. It also defines a class Program with a Main method that prints the value of x.
- Terminal:** PowerShell tab showing PS D:\parameters> dotnet run
- Status Bar:** In 8. Col 9, Spaces: 4, UTF-8 with BOM, CRLF, 08:50, 24-08-2021

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

```
4  {
5      class Temp
6      {
7          public static readonly int x=10;
8          static Temp()
9          {
10             x+=15;
11         }
12     }
13     class Program
14     {
15         static void Main(string[] args)
16         {
17             Console.WriteLine($"{Temp.x}");
18         }
19     }
20 }
21
22
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
15
PS D:\parameters>

Ln 10, Col 17 Spaces: 4 UTF-8 with BOM CRLF C# PowerShell

Type here to search

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

```
4  {
5      class Temp
6      {
7          public static readonly int x=10;
8          static Temp()
9          {
10             x+=15;
11         }
12     }
13     class Program
14     {
15         static void Main(string[] args)
16         {
17             Console.WriteLine($"{Temp.x}");
18         }
19     }
20 }
21
22
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
15
PS D:\parameters> dotnet run
25
PS D:\parameters>

Ln 9, Col 10 Spaces: 4 UTF-8 with BOM CRLF C# PowerShell

Type here to search

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

Program.cs

```
5 class Temp
6 {
7     public static readonly int x=10;
8     public static void MyMethod()
9     {
10        x = 15; I
11    }
12 }
13 class Program
14 {
15     static void Main(string[] args)
16     {
17         Console.WriteLine($"{Temp.x}");
18         Temp.MyMethod();
19         Console.WriteLine($"{Temp.x}");
20     }
21 }
22 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run

D:\parameters\Program.cs(10,13): error CS0198: A static readonly field cannot be assigned to (except in a static constructor or a variable initializer) [D:\parameters\parameters.csproj]

The build failed. Fix the build errors and run again.

PS D:\parameters>

Type here to search

Ln 10, Col 20 (9 selected) Spaces: 4 UTF-8 with BOM CRLF C# 🇺🇸

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

Program.cs

```
7 public readonly int x=10;
8 public void MyMethod()
9 {
10    x = 15;
11 }
12 class Program
13 {
14     static void Main(string[] args)
15     {
16         Temp t = new Temp();
17         Console.WriteLine($"{t.x}");
18         t.MyMethod();
19         Console.WriteLine($"{t.x}");
20     }
21 }
22 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run

D:\parameters\Program.cs(10,13): error CS0191: A readonly field cannot be assigned to (except in a constructor or init-only setter of the type in which the field is defined or a variable initializer) [D:\parameters\parameters.csproj]

The build failed. Fix the build errors and run again.

PS D:\parameters>

Type here to search

Ln 17, Col 33 Spaces: 4 UTF-8 with BOM CRLF C# 🇺🇸

VS 10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

```
Program.cs
4  {
5      class Temp
6      {
7          public readonly void MyMethod()
8          {
9              Console.WriteLine("inside mymethod...");
10         }
11     }
12   class Program
13   {
14       static void Main(string[] args)
15       {
16           Temp t = new Temp();
17           t.MyMethod();
18       }
19   }
20 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
D:\parameters\Program.cs(7,30): error CS0106: The modifier 'readonly' is not valid for this item [D:\parameters\parameters.csproj]
The build failed. Fix the build errors and run again.
PS D:\parameters>

Ln 17, Col 26 Spaces: 4 UTF-8 with BOM CRLF C#

Type here to search

VS 10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER
PARAMETERS
> bin
> obj
parameters.csproj
Program.cs

```
Program.cs
1  using System;
2
3  namespace parameters
4  {
5      class Temp
6      {
7          public const int x = 10;
8      }
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             //Temp t = new Temp();
14             Console.WriteLine($"{Temp.x}");
15         }
16     }
17 }
18
19
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
10
PS D:\parameters>

Ln 13, Col 35 Spaces: 4 UTF-8 with BOM CRLF C#

Type here to search

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - Visual Studio Code

EXPLORER
PARAMETERS
bin
obj
parameters.csproj
Program.cs

```
Program.cs
1 using System;
2
3 namespace parameters
4 {
5     class Temp
6     {
7         public static const int x = 10;
8     }
9     class Program
10    {
11        static void Main(string[] args)
12        {
13            //Temp t = new Temp();
14            Console.WriteLine($"{Temp.x}");
15        }
16    }
17 }
18
19
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
D:\parameters\Program.cs(7,33): error CS0504: The constant 'x' cannot be marked static [D:\parameters\parameters.csproj]
The build failed. Fix the build errors and run again.
PS D:\parameters>

Ln 7, Col 23 Spaces: 4 UTF-8 with BOM CRLF C#

Type here to search

10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer

Home Insert Page Layout References Mailings Review View Design Layout

Document1 - Microsoft Word

Table Tools

Constant	readonly
“const” keyword is used to declare variable as constant	“readonly” keyword is used to declare variable as readonly
The value is determined at compile time	Value is determined at runtime.
It can be initialized only at the time of declaration	It can be initialized at the time declaration as well as inside constructor
Constant are only non-static	Readonly can be both static / non-static

Words: 66

Type here to search

Document1 - Microsoft Word

Constant	readonly
"const" keyword is used to declare variable as constant	"readonly" keyword is used to declare variable as readonly
The value is determined at compile time	Value is determined at runtime.
It can be initialized only at the time of declaration	It can be initialized at the time declaration as well as inside constructor
Constant are only non-static	Readonly can be both static / non-static
Constants are accessible by classname	readonly are accessible by both classname or instance name depending on it is static/non-static

Program.cs - parameters - Visual Studio Code

```

1  namespace parameters
2  {
3      class MyClass
4      {
5          int Var1 =10;
6          public int ReturnMaxSum(int Var1)
7          {
8              return Var1 > this.Var1 ? Var1 : this.Var1;
9          }
10     class Program
11     {
12         static void Main(string[] args)
13         {
14             MyClass mc1 = new MyClass();
15             Console.WriteLine($"{mc1.ReturnMaxSum(30)}");
16             MyClass mc2 = new MyClass();
17             Console.WriteLine($"{mc2.ReturnMaxSum(5)}");
18         }
19     }
20 }
21 
```

PS D:\parameters>

Rectangular Array

A screenshot of Visual Studio Code showing a C# program named Program.cs. The code defines a class Program with a Main method that creates a 2x3 integer array and prints its elements to the console. The output window shows the resulting 2x3 matrix.

```
class Program
{
    static void Main(string[] args)
    {
        int[,] a = new int[2,3] {
            {1,2,3},
            {4,5,6}
        };

        for (int i=0;i<2;i++)
        {
            for(int j=0;j<3;j++)
            {
                Console.Write($"{a[i,j]} ");
            }
            Console.WriteLine();
        }
    }
}
```

Output:

```
PS D:\parameters> dotnet run
PS D:\parameters> dotnet run
1 2 3
4 5 6
PS D:\parameters> []
```

A screenshot of Visual Studio Code showing a modified C# program named Program.cs. The code now reads three values from the console and initializes the first row of a 2x3 array. The output window shows the user input and the resulting 2x3 matrix.

```
static void Main(string[] args)
{
    int[,] a = new int[2,3] ;

    for (int i=0;i<2;i++)
    {
        for(int j=0;j<3;j++)
        {
            Console.Write("Enter value : ");
            a[i,j] = Convert.ToInt32(Console.ReadLine());
        }
        Console.WriteLine();
    }

    for (int i=0;i<2;i++)
    {
        for(int j=0;j<3;j++)
        {
            Console.Write($"{a[i,j]} ");
        }
        Console.WriteLine();
    }
}
```

Output:

```
Enter value : 4
Enter value : 5
Enter value : 6
1 2 3
4 5 6
PS D:\parameters> []
```

The screenshot shows a Windows desktop environment with a Visual Studio Code window open. The title bar reads "VS 10.9.2.82 (DESKTOP-ZONU3B3) - VNC Viewer". The code editor displays a C# file named "Program.cs" with the following content:

```
static void Main(string[] args)
{
    // int[,] a = new int[2,3] ;
    // for (int i=0;i<2;i++)
    // {
    //     for(int j=0;j<3;j++)
    //     {
    //         Console.Write("Enter value : ");
    //         a[i,j] = Convert.ToInt32(Console.ReadLine());
    //     }
    //     Console.WriteLine();
    // }
    int[] a = new int[3]{1,2,3};
    foreach (int i in a)
    {
        Console.Write($"{i} ");
    }
    // for (int i=0;i<2;i++)
    // {
    //     for(int j=0;j<3;j++)
    //     {
    // 
```

The terminal below shows the command "dotnet run" being executed, followed by the output "1 2 3".

VS Code status bar details: Ln 25, Col 14 (133 selected), Spaces: 4, UTF-8 with BOM, CRLF, C#, 09:08, 26-08-2021.

The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-2ONU3B3) - VNC Viewer
- Menu Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Code Editor:** Program.cs - parameters - Visual Studio Code. The code defines a 3x3 matrix and reads values from the user.

```
static void Main(string[] args)
{
    int[,] a = new int[2,3] ;

    for (int i=0;i<2;i++)
    {
        for(int j=0;j<3;j++)
        {
            Console.Write("Enter value : ");
            a[i,j] = Convert.ToInt32(Console.ReadLine());
        }
        Console.WriteLine();
    }
    foreach (int i in a)
    {
        Console.Write($"{i} ");
    }

    // for (int i=0;i<2;i++)
    // {
    //     for(int j=0;j<3;j++)
    //     {
    
```

- Terminal:** powershell. It shows the command `dotnet run` being run and the output of three user inputs: "1", "2", and "3".

```
PS D:\parameters> dotnet run
Enter value : 1
Enter value : 2
PS D:\parameters> dotnet run
Enter value : 1
Enter value : 2
Enter value : 3
```

- Status Bar:** In 32, Col 34, Spaces: 4, UTF-8 with BOM, CRLF, C, 09:10, 26-08-2021, ENG
- Bottom Bar:** Type here to search, Taskbar icons (File Explorer, File Manager, Edge, Google Chrome, File Explorer, File Manager, File Explorer).

10.9.2.82 (DESKTOP-ZONUJ383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER PARAMETERS > bin > obj parameters.csproj Program.cs

Program.cs

```
5 class Program
6 {
7     static void Main(string[] args)
8     {
9         int[,] a = new int[2,3] {
10            {1,2,3},
11            {2,3,4}
12        };
13        Console.WriteLine($"{a.Length},{a.Rank}");
14    }
15 }
16 }
17 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
6,2
PS D:\parameters>

powerShell + v x

Ln 15, Col 1 Spaces: 4 UTF-8 with BOM CRLF C# AP

Type here to search

Outline

0 0 △ 0

Type here to search

Windows Taskbar: File Explorer, Edge, Google Chrome, Visual Studio Code, FileZilla, Task View

Ln 15, Col 1 Spaces: 4 UTF-8 with BOM CRLF C# AP 09:12 26-08-2021

powerShell + v x

Ln 15, Col 1 Spaces: 4 UTF-8 with BOM CRLF C# AP 09:12 26-08-2021

10.9.2.82 (DESKTOP-ZONUJ383) - VNC Viewer

File Edit Selection View Go Run Terminal Help

Program.cs - parameters - Visual Studio Code

EXPLORER PARAMETERS > bin > obj parameters.csproj Program.cs

Program.cs

```
5 class Program
6 {
7     static void Main(string[] args)
8     {
9         int[,] a = new int[2,3] {
10            {1,2,3},
11            {2,3,4}
12        };
13        Console.WriteLine($"{a.Length},{a.Rank}");
14        Console.WriteLine($"{a.GetLength(0)}");
15        Console.WriteLine($"{a.GetLength(1)}");
16    }
17 }
18 }
19 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS D:\parameters> dotnet run
6,2
2
3
PS D:\parameters>

powerShell + v x

Ln 16, Col 48 Spaces: 4 UTF-8 with BOM CRLF C# AP

Type here to search

Outline

0 0 △ 0

Type here to search

Windows Taskbar: File Explorer, Edge, Google Chrome, Visual Studio Code, FileZilla, Task View

Ln 16, Col 48 Spaces: 4 UTF-8 with BOM CRLF C# AP 09:13 26-08-2021

powerShell + v x

Ln 16, Col 48 Spaces: 4 UTF-8 with BOM CRLF C# AP 09:13 26-08-2021

The screenshot shows two instances of Visual Studio Code running side-by-side. Both instances have the same code editor window open, displaying a C# program named Program.cs. The code defines a class Program with a Main method that prints a 2x3 jagged array to the console.

```
class Program
{
    static void Main(string[] args)
    {
        int[, ] a = new int[2,3]
        {
            {1,2,3},
            {2,3,4}
        };
        for (int i=0; i<a.GetLength(0); i++)
        {
            for (int j=0; j<a.GetLength(1); j++)
            {
                Console.WriteLine($"{a[i,j]}");
            }
            Console.WriteLine();
        }
        // Console.WriteLine($"{a.Length},{a.Rank}");
        // Console.WriteLine($"{a.GetLength(0)}");
        // Console.WriteLine($"{a.GetLength(1)}");
    }
}
```

The terminal below the code editor shows the output of the dotnet run command:

```
PS D:\parameters> dotnet run
123
234
```

The status bar at the bottom right of each instance shows the date and time as 26-08-2021 and 09:15.

Jagged Array

The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-2ONU3B3) - VNC Viewer
- Menu Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Code Editor:** Program.cs - parameters - Visual Studio Code. The code defines a class Program with a Main method that creates a 2D integer array a of size 2x3, initializes its first row to size 3, and prints the lengths of both rows.
- Terminal:** Shows the command PS D:\parameters> dotnet run being run twice, resulting in the output 2,1 and 3,2 respectively.
- Status Bar:** Lines 14, Col 1 (94 selected), Spaces: 4, UTF-8 with BOM, CRLF, and various icons for file operations and status.
- Bottom Navigation:** Type here to search, taskbar with icons for File Explorer, Task View, Edge, Google Chrome, File Explorer, and VS Code.

The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer
- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Editor:** Program.cs - parameters - Visual Studio Code
- Code Content:**

```
14     for (int i=0;i<a.Length;i++)
15     {
16         for (int j=0;j<a[i].Length;j++)
17         {
18             Console.WriteLine("Enter value : ");
19             a[i][j] = Convert.ToInt32(Console.ReadLine());
20         }
21     }
22     for (int i=0;i<a.Length;i++)
23     {
24         for (int j=0;j<a[i].Length;j++)
25         {
26             Console.WriteLine($"{a[i][j]}");
27         }
28         Console.WriteLine();
29     }
30 //    Console.WriteLine($"(a.Length),(a.Rank)");
31 //    Console.WriteLine($"{a[0].Length},{a[1].Length}");
32 }
33 }
```
- Bottom Status Bar:** PS D:\parameters> dotnet run
Enter value : 2
Enter value : 1
Enter value : 2
Enter value : 3
Enter value : 4
212
PS D:\parameters> []
- Terminal Tab:** powershell +v
- Bottom Icons:** Windows Start, Task View, Taskbar icons (File Explorer, Edge, Google Chrome, File Manager, Task View, Taskbar search).

The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** 10.9.2.82 (DESKTOP-ZONU383) - VNC Viewer
- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Editor:** Program.cs - parameters - Visual Studio Code
- Code Content:**

```
6     class Program
7     {
8         static void Main(string[] args)
9         {
10            Console.WriteLine("How many rows? : ");
11            int rows = Convert.ToInt32(Console.ReadLine());
12            int[][] a = new int[rows][];
13            int m;
14            for (int i=0;i<rows;i++)
15            {
16                Console.WriteLine($"Enter no. of elements for {i+1} Row : ");
17                m = Convert.ToInt32(Console.ReadLine());
18                a[i] = new int[m];
19            }
20            for (int i=0;i<a.Length;i++)
21            {
22                Console.WriteLine($"Enter input for {i+1} Row");
23                for (int j=0;j<a[i].Length;j++)
24                {
25                    Console.WriteLine("Enter value : ");
26                    a[i][j] = Convert.ToInt32(Console.ReadLine());
27                }
28            }
29            for (int i=0;i<a.Length;i++)
30            {
31                for (int j=0;j<a[i].Length;j++)
32                {
33                    Console.WriteLine($"{a[i][j]} ");
34                }
35                Console.WriteLine();
36            }
37        }
38 }
```
- Bottom Status Bar:** Ln 17, Col 17 Spaces: 4 UTF-8 with BOM CRLF 10:17 26-08-2021 []
PS D:\parameters> []
- Terminal Tab:** powershell +v
- Bottom Icons:** Windows Start, Task View, Taskbar icons (File Explorer, Edge, Google Chrome, File Manager, Task View, Taskbar search).

Write a C# Program to create 3x4 int array Get the nos from user and print the same.

Write a C# Program to create a jagged array of size 3 get the no of elements for each subarray from user and take the values from user and print the same