

## LINQ Interview Questions and Answers

[What is Language Integrated Query \(LINQ\)?](#)

Answer :: Language Integrated Query (LINQ) adds the ability to query objects using .NET languages. The LINQ to SQL object/relational mapping (O/RM) framework provides the following basic features:

- Tools to create classes (usually called entities) mapped to database tables

- Compatibility with LINQ's standard query operations

- The DataContext class, with features such as entity record monitoring, automatic SQL statement generation, record concurrency detection, and much more

.... [Read More](#)

[What are the main difference between LINQ and Stored Procedures?](#)

Answer :: There are couple of advantage of LINQ over stored procedures.

1. Debugging - It is really very hard to debug the Stored procedure but as LINQ is part of .NET, you can use visual studio's debugger to debug the queries.
2. Deployment - With stored procedures, we need to provide an additional script for stored procedures but with LINQ everything gets compiled into single DLL hence deployment becomes easy.
3. Type Safety - LINQ is type safe, so queries errors are type checked at compile time. It is really good to encounter an error when compiling rather than runtime exception! .... [Read More](#)

[What are the pros and cons of LINQ \(Language-Integrated Query\)?](#)

[What are the best and worst cases in which to use LINQ?](#)

[How have you benefitted or not benefitted from using LINQ?](#)

[Which data sources benefit the least and the most from LINQ?](#)

Answer :: Pros:

- Declarative approach makes queries easier to understand and more compact

- Extensibility and expression trees allow mostly consistent querying of multiple sources

- Even in-process queries can be implemented in ways other than LINQ to Objects - e.g. Parallel LINQ and my own Push LINQ framework. Very flexible.

- Fabulously useful for in-process queries, where it's easiest to understand

Great to be able to avoid SQL in strings etc

Wide range of operators provided by default, and others can easily be added for LINQ to Objects

Language features introduced primarily for LINQ are widely applicable elsewhere (yay for lambdas)

Cons:

Query expressions aren't understood well enough, and are overused. Often simple method invocation is shorter and simpler.

Inevitable inconsistencies between provider - impedance mismatch is still present, which is reasonable but needs to be understood

There will always be some things you can do in SQL but not in LINQ

Without understanding what's going on, it's easy to write very inefficient code

It's hard to write a LINQ provider. This may well be inevitable, but more guidance from Microsoft would be appreciated.

It's a new way of thinking about data access for most developers, and will need time for understanding to percolate

Not specifically LINQ but related to it - the way extension methods are discovered in C# isn't granular enough

Some operators are "missing", particularly the equivalents of `OrderBy` for things other than ordering - e.g. finding the item with the maximum value of a property

Deferred execution and streaming are poorly understood (but improving)

Debugging can be very tricky due to deferred execution and streaming

I find it's best when dealing with in-process queries. They're easy to predict, understand and extend.

Complementary technologies like LINQ to XML and Parallel LINQ are great. LINQ to Objects can be used almost anywhere.

LINQ to SQL etc are really nice where they're appropriate, but they're harder to understand and need more expertise. They're also only applicable in certain areas of your code. .... [Read More](#)

[Can you tell me about the disadvantages of LINQ over Stored Procedures?](#)

Answer ::

Network traffic: sprocs need only serialize sproc-name and argument data over the wire while LINQ sends the entire query. This can get really bad if the queries are very complex. However, LINQ's abstraction allows Microsoft to improve this over time.

Less flexible: Sprocs can take full advantage of a database's featureset. LINQ tends to be more generic in it's support. This is common in any kind of language abstraction (e.g. C# vs assembler).

Recompiling: If you need to make changes to the way you do data access, you need to recompile, version, and redeploy your assembly. Sprocs can sometimes allow a DBA to tune the data access routine without a need to redeploy anything.

.... [Read More](#)

[Can you tell me some advantages of LINQ over Stored Procedures?](#)

Answer :: For basic data retrieval I would be going for Linq without hesitation.



Perhaps that is the cause because it things it needs to create y? .... [Read More](#)

[i,m in doubt to choose which one from .net java. i need perfect definition about j2se , j2ee, j2me, jsp, jdbc , c#, asp.net, ado.net , linq and their differences. would u help me in this dilemma? i realy need this. by the way i can program by c++ and vb.thank u.](#)

Answer :: You should use Java.

.net only works on Microsoft machines, Java is platform independent. Plus, it is very similar to C++. .... [Read More](#)

[trying to call paint method](#)

[here is the code I have so far](#)

```
-  
using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Windows.Forms;  
  
-  
public partial class Form1 : Form  
{  
    public class Snake  
    {  
        static void Main\(\) { }  
    }  
        //somewhat of a coordinate system for the game  
        bool\[,\] point = new bool\[51, 51\];  
        //sets the direction that the snake is supposed to move: 1:up, 2:down, 3:left, 4:right,  
        int direction = 4;  
        public Form1\(\)  
        {  
            InitializeComponent\(\);  
            //This sets the starting snake  
            point\[0, 0\] = true;  
        }  
  
-  
        private void InitializeComponent\(\)  
        {  
            throw new NotImplementedException\(\);  
        }
```

```
//This method is necessary because InitializeComponent() calls it
```

```
//this.Load += new System.EventHandler(this.Form1_Load);
```

```
private void Form1_Load(object sender, EventArgs e)
```

```
{
```

```
}
```

```
private void Form1_Paint(object sender, PaintEventArgs e)
```

```
{
```

```
//according to the array point this displays the blocks of the snake and fruit
```

```
Graphics g = e.Graphics;
```

```
int x = 0; int y = 0;
```

```
while (y < 51)
```

```
{
```

```
bool skip = false;
```

```
if (point[x, y] == true)
```

```
{ g.FillRectangle(Brushes.GreenYellow, x * 10, y * 10, 10, 10); }
```

```
if (x >= 50) { x = 0; y++; skip = true; }
```

```
if (x < 50 skip == false) { x++; }
```

```
}
```

```
}
```

```
private void timer1(object sender, EventArgs e)
```

```
{
```

```
try
```

```
{
```

```
int x = 0; int y = 0;
```

```
while (y < 51)
```

```
{
```

```
bool skip = false;
```

```
if (direction == 4)
```

```
{ if (point[x, y] == true) { point[x, y] = false; point[x + 1, y] = true; break; } }
```

```
if (direction == 3)
```

```
{ if (point[x, y] == true) { point[x, y] = false; point[x - 1, y] = true; break; } }
```

```
if (direction == 2)
```

```
{ if (point[x, y] == true) { point[x, y] = false; point[x, y + 1] = true; break; } }
```

```
if (direction == 1)
```

```
{ if (point[x, y] == true) { point[x, y] = false; point[x, y - 1] = true; break; } }
```

```
if (x >= 50) { x = 0; y++; skip = true; }
```

```
if (x < 50 skip == false) { x++; }
```

```
}
```

```
Invalidate();
```

```
}
```

```

catch
{
    MessageBox.Show("Game Over");
}
}

-
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    // Invalidate the Eater before moving it
    string result = e.KeyData.ToString();
    switch (result)
    {
        case "Left":
            direction = 3; break;
        case "Right":
            direction = 4; break;
        case "Up":
            direction = 1; break;
        case "Down":
            direction = 2; break;
    }
}
}
}
-

```

Answer :: You don't really call paint; you can either overload it or attach to it. That way it's automatically called when the paint event is fired.

Overloading, put this in your snake class:

```

protected override void OnPaint(PaintEventArgs e)
{
    // Call the base OnPaint event to make sure the rest of the form gets drawn.
    base.OnPaint(e);

    // Do your drawing here.
}

```

Attaching to the event:

Put this in your initializer:

```
this.Paint += new System.Windows.Forms.PaintEventHandler ( YourFunction_Paint );
```

Then add your function that does the drawing to the form:

```
private void YourFunction_Paint(object sender, PaintEventArgs e)
{
    // Put your drawing code here
} .... Read More
```

I am using the User.Identity.Name property to display my users name when they comment on things on my site, the problem is that User.Identity.Name works using data they entered at Login, so their names are hardly ever capitalized as they should be.

Now I had been using an SQL call to access their proper names, but that is processor overhead that I don't want.

Is there a way that I can do a single SQL call when they log on (in code-behind) that would reset the User.Identity.Name to the proper caps (as entered in my db)? I just need a code sample or something.

I am using MS SQL and I don't want to mess with LINQ, I am too far into the development cycle.

Answer :: You can simply store the retrieved value in a Session variable and read from the Session variable.

First time you read the username (when a user logs in):

```
Session["UserName"] = User.Identity.Name;
```

//later when you want to retrieve the value you can use:

```
string UserName;
if(Session["UserName"] != null) {
    UserName = Session["UserName"].ToString();
}
```

Hope this helps. .... [Read More](#)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Drawing.Drawing2D;
```

```
-
namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
```

```

- {
-   public Form1()
-   {
-       InitializeComponent();
-   }
-
-   private void onPaint(object sender, PaintEventArgs e)
-   {
-       Graphics g = e.Graphics;
-
-       Brush yel = new SolidBrush(Color.Yellow);
-       Brush grn = new SolidBrush(Color.Green);
-       Brush blu = new SolidBrush(Color.SkyBlue);
-
-       Rectangle sun = new Rectangle(Width-100,15 , 80, 80);
-       Rectangle grass = new Rectangle(0, 300, Width, Height);
-       Rectangle sky = new Rectangle(0, 0, Width, Height);
-
-       g.FillRectangle(blu, sky);
-       g.FillRectangle(grn, grass);
-       g.FillEllipse(yel, sun);

```

Answer :: A simple way would be to break paint out into its own function (with a graphics object as a parameter) then call it with either e.Graphics (from onPaint) or with Form1.CreateGraphics() (from whatever function). ....

[Read More](#)

[I have this code completed so far but i cannot get anything to pop up when i build solution... I will include the code and if anyone can help me to get output I would greatly appreciate it...](#)

```

- Form1.cs
- using System;
- using System.Collections.Generic;
- using System.ComponentModel;
- using System.Data;
- using System.Drawing;
- using System.Text;
- using System.Windows.Forms;
-
- namespace Snakin
- {
-   public partial class Form1 : Form
-   {
-       //somewhat of a coordinate system for the game

```

```
bool[,] point = new bool[51, 51];
```

```
//sets the direction that the snake is supposed to move: 1:up, 2:down, 3:left, 4:right,
```

```
int direction = 4;
```

```
public Form1()
```

```
{
```

```
    InitializeComponent();
```

```
    //This sets the starting snake
```

```
    point[0, 0] = true;
```

```
}
```

```
private void Form1_Paint(object sender, PaintEventArgs e)
```

```
{
```

```
    //according to the array point this displays the blocks of the snake and fruit
```

```
    Graphics g = e.Graphics;
```

```
    int x = 0; int y = 0;
```

```
    while (y < 51)
```

```
    {
```

```
        bool skip = false;
```

```
        if (point[x, y] == true)
```

```
            { g.FillRectangle(Brushes.GreenYellow, x * 10, y * 10, 10, 10); }
```

```
        if (x >= 50) { x = 0; y++; skip = true; }
```

```
        if (x < 50 && skip == false) { x++; }
```

```
    }
```

```
}
```

```
private void Form1_KeyDown(object sender, KeyEventArgs e)
```

```
{
```

```
    // Invalidate the Eater before moving it
```

```
    string result = e.KeyData.ToString();
```

```
    switch (result)
```

```
    {
```

```
        case "Left":
```

```
            direction = 3; break;
```

```
        case "Right":
```

```
            direction = 4; break;
```

```
        case "Up":
```

```
            direction = 1; break;
```

```
        case "Down":
```

```
            direction = 2; break;
```

```
    }
```

```
}
```

```

-
private void timer1_Tick_1(object sender, EventArgs e)
{
    try
    {
        int x = 0; int y = 0;
        while (y < 51)
        {
            bool skip = false;
            if (direction == 4)
                { if (point[x, y] == true) { point[x, y] = false; point[x + 1, y] = true; break; } }
            if (direction == 3)
                { if (point[x, y] == true) { point[x, y] = false; point[x - 1, y] = true; break; } }
            if (direction == 2)
                { if (point[x, y] == true) { point[x, y] = false; point[x, y + 1] = true; break; } }
            if (direction == 1)
                { if (point[x, y] == true) { point[x, y] = false; point[x, y - 1] = true; break; } }
            if (x >= 50) { x = 0; y++; skip = true; }
            if (x < 50 skip == false) { x++; }
        }
        Invalidate();
    }
    catch
    {
        MessageBox.Show("Game Over");
    }
}
}
}
}

```

-  
Program.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

```

-  
namespace Snakin

```

{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
    }
}

```

```
/// &lt;/summary&gt;
```

```
[STAThread]
```

```
static void Main()
```

```
{
```

```
    Application.EnableVisualStyles();
```

```
    Application.SetCompatibleTextRenderingDe...
```

```
    Application.Run(new Form1());
```

```
}
```

```
}
```

```
}
```

Answer :: Try putting `timer1.Start()` at the end of the `Form1` constructor. If that doesn't fix the problem I need to see your `Form1.Designer.cs`. .... [Read More](#)

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.ComponentModel;
```

```
using System.Data;
```

```
using System.Drawing;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Windows.Forms;
```

```
using System.IO;
```

```
namespace pipeworksSlidingPuzzle {
```

```
    public partial class mainWindow : Form {
```

```
        int rows, cols, blank, r, c;
```

```
        String path = "F:\\IT312\\pipeworksSlidingPuzzle\\pipe...
```

```
        int index = 0, count = 0;
```

```
        Button b1;
```

```
        bool secondClick = false;
```

```
        bool shuffling;
```

```
        int tileWidth, tileHeight;
```

```
    public mainWindow()
```

```
    {
```

```
        InitializeComponent();
```

```
    }
```

```
    private void Form1_Load(object sender, EventArgs e)
```

```
    {
```

```
        AutoSize = true;
```

```
-
```

```
String[] files = Directory.GetFiles(path, "*.jpg");
```

```
-  
    for (int r = 0; r < rows; r++) {  
        for (int c = 0; c < cols; c++) {  
            if (r == rows - 1 & c == cols - 1) {  
                break;  
            }  
            PictureBox pb = new PictureBox();  
            pb.Size = new Size(100, 100);  
            pb.SizeMode = PictureBoxSizeMode.StretchImage;  
            pb.Margin = new Padding(0);  
            pb.BorderStyle = BorderStyle.FixedSingle;  
            pb.Location = new Point(c * 100, r * 100);  
            pb.Image = Image.FromFile(files[index]);  
            pb.Tag = index;  
            count++;  
            if (count==2)  
            {  
                index++;  
                count = 0;  
            }  
            Controls.Add(pb);
```

```
-  
        Button b = new Button();  
        b.Location = pb.Location;  
        b.Size = pb.Size;  
        b.Tag = pb;  
        Controls.Add(b);  
        b.BringToFront();  
        b.Click += new EventHandler(ButtonClick);
```

```
    }  
    }  
    blank = rows * cols - 1;
```

```
    private void Shuffle() {  
        // shuffle...  
        shuffling = true;  
        Random rnd = new Random();  
        for (int i = 0; i < 1000; i++) {  
            int x = rnd.Next(rows * cols - 1) + 1;  
            pb_Click(Controls[x], null);  
        }  
        shuffling = false;
```

```

Refresh();
}
void pb_Click(object sender, EventArgs e) {
int brow = blank / cols;
int bcol = blank % rows;

PictureBox pb = (PictureBox) sender;
object o = pb.Tag;
int tileIndex = (int) o;

int trow = tileIndex / cols;
int tcol = tileIndex % rows;

// down...
if (tcol == bcol trow + 1 == brow) {
if (shuffling) {
pb.Top += tileHeight;
} else {
for (int i = 0; i < tileHeight; i++) {
pb.Top++;
}
}
blank = tileIndex;
tileIndex += cols;
pb.Tag = tileIndex;
if (isSolved()) {
MessageBox.Show("nalpas!!!");
}
return;
}

// up...
if (tcol == bcol trow - 1 == brow) {
if (shuffling) {
pb.Top -= tileHeight;
} else {
for (int i = 0; i < tileHeight; i++) {
pb.Top--;
}
}
blank = tileIndex;
tileIndex -= cols;
pb.Tag = tileIndex;
if (isSolved()) {

```

```
    MessageBox.Show("nalpas!!!");
```

```
    }
```

```
    return;
```

```
-  
    // left...
```

```
    if (tcol - 1 == bcol  trow == brow) {
```

```
        if (shuffling) {
```

```
            pb.Left -= tileWidth;
```

```
        } else {
```

```
            for (int i = 0; i < tileHeight; i++) {
```

```
                pb.Left--;
```

```
            }
```

```
        }
```

```
        blank = tileIndex;
```

```
        tileIndex--;
```

```
        pb.Tag = tileIndex;
```

```
        if (isSolved()) {
```

```
            MessageBox.Show("nalpas!!!");
```

```
        }
```

```
    return;
```

```
-  
    // right...
```

```
    if (tcol + 1 == bcol  trow == brow) {
```

```
        if (shuffling) {
```

```
            pb.Left += tileWidth;
```

```
        } else {
```

```
            for (int i = 0; i < tileHeight; i++) {
```

```
                pb.Left++;
```

```
            }
```

```
        }
```

```
        blank = tileIndex;
```

```
        tileIndex++;
```

```
        pb.Tag = tileIndex;
```

```
        if (isSolved())
```

Answer :: The syntax is wrong. Please check the syntax. .... [Read More](#)

Hello fellow programmers!

-  
I originally did an undergraduate in mathematics and the only 'coding' software development I did was in Matlab, and VBA (Excel) and to a smaller extent C - but purely from a procedural perspective. I went out and worked for a number of years as a Quantitative Analyst for a Financial Software Development Team and decided to go back to uni to learn Scientific

Software Development, which I love; but I'm finding it really difficult choosing a language that I want to specialise in, I have to admit on the surface Java and C# are appealing as they are so strongly typed, however I really enjoy the flexibility and absolute control that C++, Fortran and Ada offer - but in terms of the work involved in application development I have noticed a severe increase in the amount of time required.

-  
Is C++ worth the extra work, or am I better off sticking with Java and C# ?

I plan on doing alot of Form application development involving LINQ and Access/SQL Server communication

-  
Any guidance would be greatly appreciated,

-  
David

Answer :: I think that if you are donig scientific work, with large or long running data sets, then C++ is very useful. Even with all the JIT improvements, C++ still runs faster and in less memory than Java or even C#.

However, for projects that include a lot of UI work, C# and Java are better. the libraries are significantly larger. For the form and database work, C# and Java are significantly better. .... [Read More](#)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

-  
namespace ConsoleApplication30

{

class Program

{

static void Main(string[] args)

{

string userIn = "0";

while (userIn != "e")

{

Console.WriteLine("Please Select an Option");

Console.WriteLine();

Console.WriteLine("Enter n to display name and student number");

Console.WriteLine("Enter t to enter the times tables.");

Console.WriteLine("Enter e to exit");

userIn = Console.ReadLine();

-

-

\_\_\_\_\_

-

-





-  
-  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
-

Answer :: I think that the following does what you have in mind.

Always be careful with your curly braces { and }. If you write  
if (cond)

always start with { and } after the condition. The same for "else if (cond)": use { and }. Same with while (cond {}). If you use the free Visual Studio Express 2008 from Microsoft, you will see the matching parens or braces highlighted. The code will also be indented properly to reflect the logic. Have fun with C#!

```
//----- begin
namespace ConsoleApplication30
{
    class Program
    {
        static void Main(string[] args)
        {
            string userIn = "0";

            while (userIn != "e")
            {
                Console.WriteLine("Please Select an Option");
                Console.WriteLine();
                Console.WriteLine("Enter n to display name and student number");
                Console.WriteLine("Enter t to enter the times tables.");
                Console.WriteLine("Enter e to exit");
                userIn = Console.ReadLine();

                if (userIn == "n")
                {
                    NameMeth();
                }
                else if (userIn == "t")
                {
                    TimeMeth();
                }
            }
        }
    }
}
```

```

else if (userIn != "e")
{
    Console.WriteLine("Error {0} is Invalid", userIn);
}
}
}

```

```

public static void NameMeth()
{
    Console.WriteLine("Name:Thomas Anderson \nStudent#: 300495656");
}

```

```

public static void TimeMeth()
{
    int x;
    int y;
    int total;
    string userIn = "0";

    Console.WriteLine("Please Enter an Integer");
    userIn = Console.ReadLine();
    y = Convert.ToInt32(userIn);

    for (x = 1; x &lt;= 12; x++)
    {
        total = y * x;
        Console.WriteLine("{0} times {1} is {2}", y, x, total);
    }
}
}
//----- end .... Read More

```

[I am trying to write a program that prompts 5 users to enter in their names and ages. the names and ages are then put into a set of arrays. after the information has been entered. the program displays the name of the oldest. I realize that I could use quite a bit of if then statements to get my program to display the oldest persons name. but there has to be an easier way. I know I could use a method as well as a loop but I could not figure it out how to do it. please assist.](#)

- [the code is listed.....](#)

-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication1
{
    class Program
    {
        public static void Main(string[] args)
        {
            int[] agenmber = new int[5];
            string[] agetxt = new string[5];
            string[] namesjumble = new string[5];

            //information on the first person

            Console.WriteLine("USER 1 INFORMATION");
            Console.WriteLine("");
            Console.WriteLine("what is your name?");
            namesjumble[0] = Console.ReadLine();
            Console.WriteLine("Hello {0}, what is your age?", namesjumble[0]);
            agetxt[0] = Console.ReadLine();
            agenmber[0] = Convert.ToInt32(agetxt[0]);
            Console.WriteLine("the first person to enter a name is {0} with an age of {1} years", namesjumble[0], agetxt[0]);

            // information for the second person

            Console.WriteLine("");
            Console.WriteLine("USER 2 INFORMATION");
            Console.WriteLine("");
            Console.WriteLine("what is your name?");
            namesjumble[1] = Console.ReadLine();
            Console.WriteLine("Hello {0}, what is your age?", namesjumble[1]);
            agetxt[1] = Console.ReadLine();
            agenmber[1] = Convert.ToInt32(agetxt[1]);
            Console.WriteLine("the second person to enter a name is {0} with an age of {1} years", namesjumble[1], agetxt[1]);

            // information for the third person

            Console.WriteLine("");
            Console.WriteLine("USER 3 INFORMATION");
            Console.WriteLine("");
```

```

Console.WriteLine("what is your name?");
namesjumble[2] = Console.ReadLine();
Console.WriteLine("Hello {0}, what is your age?", namesjumble[2]);
agetxt[2] = Console.ReadLine();
agenmber[2] = Convert.ToInt32(agetxt[2]);
Console.WriteLine("the third person to enter a name is {0} with an age of {1} years", namesjumble[2], agetxt[2]);

-
// information for the fourth person
-
Console.WriteLine("");
Console.WriteLine("USER 4 INFORMATION");
Console.WriteLine("");
Console.WriteLine("what is your name?");
namesjumble[3] = Console.ReadLine();
Console.WriteLine("Hello {0}, what is your age?", namesjumble[3]);
agetxt[3] = Console.ReadLine();
agenmber[3] = Convert.ToInt32(agetxt[3]);
Console.WriteLine("the fourth person to enter a name is {0} with an age of {1} years", namesjumble[3], agetxt[3]);

-
// information for the fifth person
-
Console.WriteLine("");
Console.WriteLine("USER 5 INFORMATION");
Console.WriteLine("");
Console.WriteLine("what is your name?");
namesjumble[4] = Console.ReadLine();
Console.WriteLine("Hello {0}, what is your age?", namesjumble[4]);
agetxt[4] = Console.ReadLine();
agenmber[4] = Convert.ToInt32(agetxt[4]);
Console.WriteLine("the fifth person to enter a name is {0} with an age of {1} years", namesjumble[4], agetxt[4]);
if (agenmber[0] > agenmber[1] agenmber[0] > agenmber[2] agenmber[0] > agenmber[3] agenmber[0]
&gt; agenmber[4])
{
    Console.WriteLine("{0} is the oldest", namesjumble[0]);
}
else if (agenmber[1] > agenmber[0] agenmber[1] > agenmber[2] agenmber[1] > agenmber[3]
agenmber[1] > agenmber[4])
{
    Console.WriteLine("{0} is the oldest", namesjumble[1]);
}
else if (agenmber[2] > agenmber[1] agenmber[2] > agenmber[0] agenmber[2] > agenmber[3]
agenmber[2] > agenmber[4])
{

```

```

    Console.WriteLine("{0} is the oldest", namesjumble[2]);
}
if (agenmber[3] > agenmber[1] agenmber[3] > agenmber[2] agenmber[3] > agenmber[0] agenmber[3]
> agenmber[4])
{
    Console.WriteLine("{0} is the oldest", namesjumble[3]);
}
if (agenmber[4] > agenmber[1] agenmber[4] > agenmber[2] agenmber[4] > agenmber[3] agenmber[4]
> agenmber[0])
{
    Console.WriteLine("{0} is the oldest", namesjumble[4]);
}
Console.ReadLine();
}

```

Answer :: More common and generic approach would be to write prompts and read input inside the loop...

```

loop (some condition){
incrCounter();
output("Enter User {0} info...");
readYourInput();
}

```

Again use the loop to get your oldest user..

```

loop (some condition){
incrCounter();
if (age[counter] > age[counter+1]){
result = counter;
}
}

```

After this loop your result will contain an index of the oldest user in the array...

.... [Read More](#)