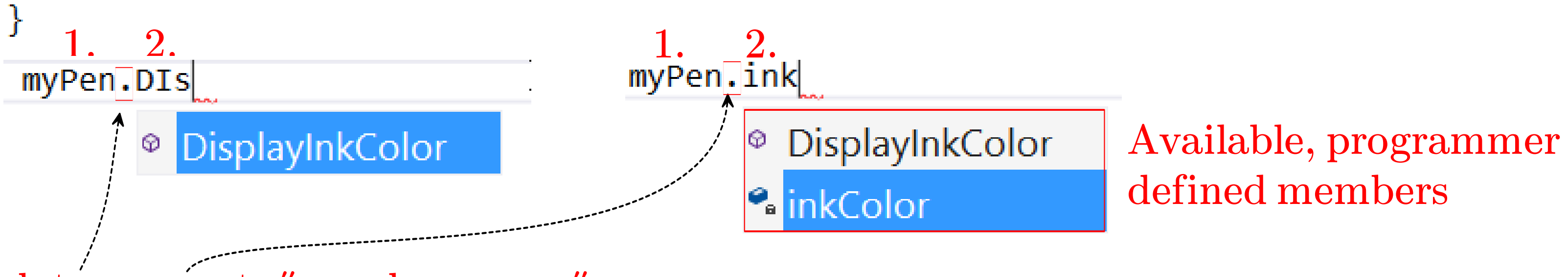
Question 74: What is member access, and how does IntelliSense work with programmer defined classes?

Answer 74: Create a simple class. As you type, features of the class should become visible in IntelliSense.

```
using static System.Console;
class Pen //1. Name of class is Pen
{
    private string inkColor;//2. Ink color field, member of Pen class
    public Pen(string ink)//3. Constructor header
    {
        inkColor= ink;//4. Sets ink color
    }
    public void DisplayInkColor() //5. Shows pen ink color, member of Pen class
    {
        WriteLine($"This pen writes in {inkColor}");
    }
    static void Main()
    {
        Pen myPen = new Pen("blue");//6. Makes new pen with blue ink
        myPen.DisplayInkColor();  This pen writes in blue
    }
}
```



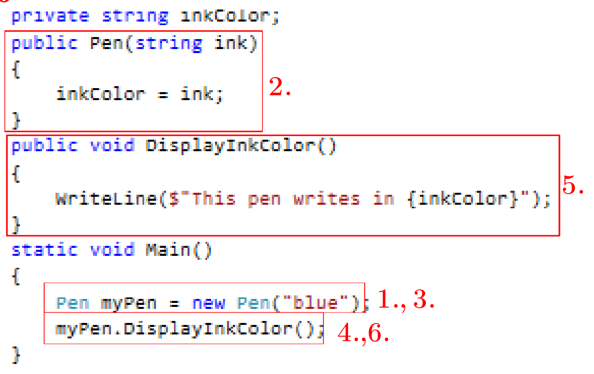
```
myPen.DIs
```

```
myPen.ink
```

dot represents "member access"

The general concept at this point is this: "name of object.name of member"

Either run the code above with the debugger so you can be 100% clear on the way it executes, or on the right is an approximate flow analysis. Follow the numbers adjacent to the boxes very carefully. Notation like 1., 3. means code begins there, goes to step 2. is, and then comes back to 3.



```
private string inkColor;
public Pen(string ink)
{
    inkColor = ink;
}
public void DisplayInkColor()
{
    WriteLine($"This pen writes in {inkColor}");
}
static void Main()
{
    Pen myPen = new Pen("blue");
    myPen.DisplayInkColor();
}
```

Check point 25: Convert the instructions below into a C# class.

1) Create a Canvas class to represent a canvas to be painted on
2) Add a string field to represent the "title" of the work
3) Add a method that can display the title of the work
4) Add a method that can draw a separator like "————————"
5) Create an instance of the of the Canvas class
6) Display the title of a canvas, and display the separator on the console screen