

Universidad Rafael Landívar

Facultad de Ingeniería.

Licenciatura En Ingeniería En Informática Y Sistemas Ingeniería

Pensamiento Computacional Matutina.

Docente: ing. Aguilar Rojas Luis Enrique.

Actividad 3, Semana 9

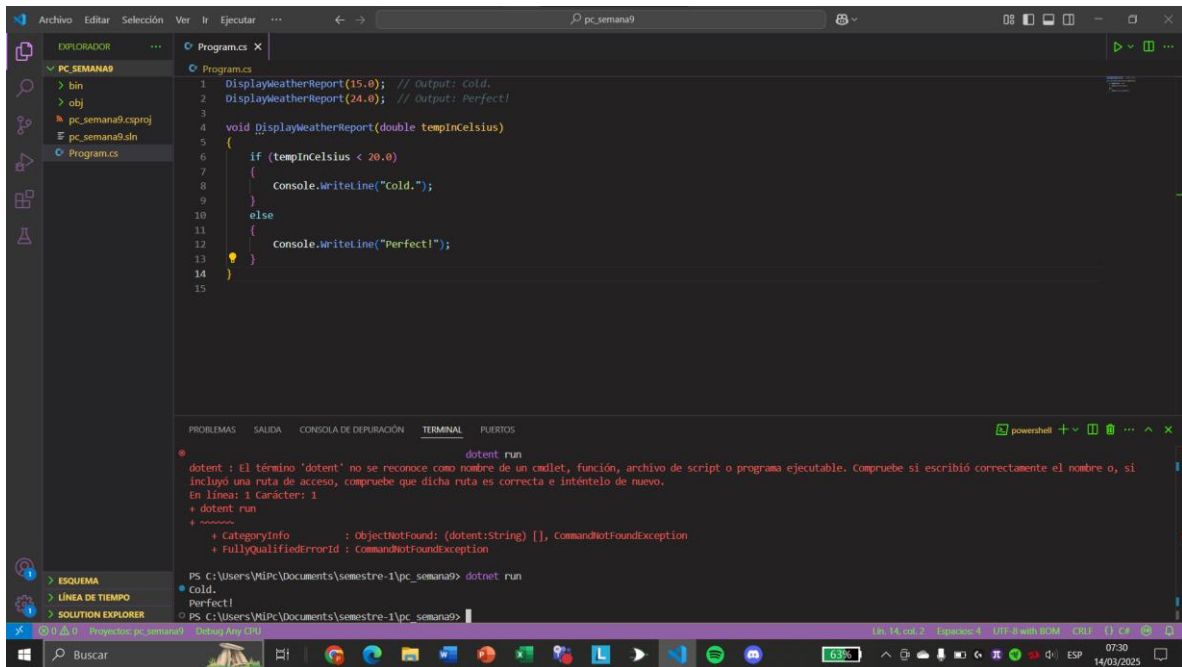
“Practica If”

Estudiante: Escobar Armas, Pablo Fernando.

Carne: 2000125.

Guatemala, 14 de marzo de 2025.

1. If

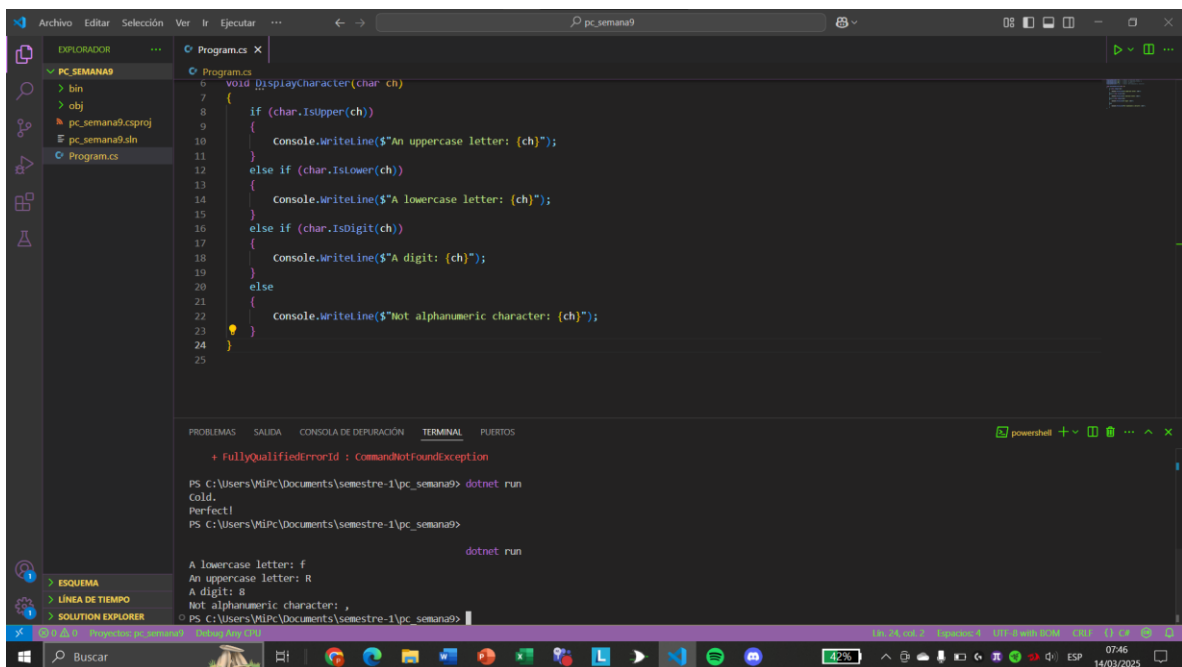


The screenshot shows the Visual Studio IDE with a C# file named Program.cs. The code defines a method `DisplayWeatherReport` that takes a `double tempInCelsius` parameter. It uses an `if` statement to check if the temperature is less than 20.0. If true, it prints "Cold."; otherwise, it prints "Perfect!". The terminal shows the command `dotnet run` being executed, resulting in the output "Cold." followed by "Perfect!".

```
1 DisplayWeatherReport(15.0); // Output: Cold.  
2 DisplayWeatherReport(24.0); // Output: Perfect!  
3  
4 void DisplayWeatherReport(double tempInCelsius)  
5 {  
6     if (tempInCelsius < 20.0)  
7     {  
8         Console.WriteLine("Cold.");  
9     }  
10    else  
11    {  
12        Console.WriteLine("Perfect!");  
13    }  
14 }  
15
```

dotnet run
dotnet : El término 'dotnet' no se reconoce como nombre de un cmdlet, función, archivo de script o programa ejecutable. Compruebe si escribió correctamente el nombre o, si
incluyo una ruta de acceso, compruebe que dicha ruta es correcta e intente de nuevo.
En línea: 1 carácter: 1
+ dotnet run
+ ~~~~~
+ CategoryInfo : ObjectNotFound: (dotnet:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
PS C:\Users\MIpc\Documents\semestre-1\pc_semana9> dotnet run
Cold.
Perfect!
PS C:\Users\MIpc\Documents\semestre-1\pc_semana9>

2. Else



The screenshot shows the Visual Studio IDE with a C# file named Program.cs. The code defines a method `DisplayCharacter` that takes a `char ch` parameter. It uses an `if-else` statement to check if the character is uppercase, lowercase, a digit, or not alphanumeric. The terminal shows the command `dotnet run` being executed, resulting in the output "A lowercase letter: f", "An uppercase letter: R", "A digit: 8", and "Not alphanumeric character: ;".

```
6 void DisplayCharacter(char ch)  
7 {  
8     if (char.IsUpper(ch))  
9     {  
10        Console.WriteLine($"An uppercase letter: {ch}");  
11    }  
12    else if (char.IsLower(ch))  
13    {  
14        Console.WriteLine($"A lowercase letter: {ch}");  
15    }  
16    else if (char.IsDigit(ch))  
17    {  
18        Console.WriteLine($"A digit: {ch}");  
19    }  
20    else  
21    {  
22        Console.WriteLine($"Not alphanumeric character: {ch}");  
23    }  
24 }  
25
```

+ FullyQualifiedErrorId : CommandNotFoundException
PS C:\Users\MIpc\Documents\semestre-1\pc_semana9> dotnet run
Cold.
Perfect!
PS C:\Users\MIpc\Documents\semestre-1\pc_semana9>
dotnet run
A lowercase letter: f
An uppercase letter: R
A digit: 8
Not alphanumeric character: ;
PS C:\Users\MIpc\Documents\semestre-1\pc_semana9>

3. Switch

The screenshot shows the Visual Studio IDE with a C# project named 'pc_semana9'. The code in 'Program.cs' defines a method `DisplayMeasurement` that uses a `switch` statement to handle different measurement values. The terminal output shows the results of running the program with various inputs.

```
1 DisplayMeasurement(-4); // Output: Measured value is -4; too low.
2 DisplayMeasurement(5); // Output: Measured value is 5.
3 DisplayMeasurement(30); // Output: Measured value is 30; too high.
4 DisplayMeasurement(double.NaN); // Output: Failed measurement.
5
6 void DisplayMeasurement(double measurement)
7 {
8     switch (measurement)
9     {
10         case < 0.0:
11             Console.WriteLine($"Measured value is {measurement}; too low.");
12             break;
13         case > 15.0:
14             Console.WriteLine($"Measured value is {measurement}; too high.");
15             break;
16         case double.NaN:
17             Console.WriteLine("Failed measurement.");
18             break;
19         default:
20             Console.WriteLine($"Measured value is {measurement}.");
21     }
22 }
```

Terminal Output:

```
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9> dotnet run
A lowercase letter: f
An uppercase letter: R
A digit: 8
Not alphanumeric character: ,
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9> dotnet run
Measured value is -4; too low.
Measured value is 5.
Measured value is 30; too high.
Failed measurement.
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9>
```

4. Bool fallo

The screenshot shows the Visual Studio IDE with a C# project named 'pc_semana9'. The code in 'Program.cs' defines a class `calaseSemanas` with a `Main` method. The terminal output shows the results of running the program. A compilation error is visible in the bottom status bar.

```
1 using System;
2 0 referencias
3 class calaseSemanas
4 {
5     0 referencias
6     static void Main()
7     {
8         bool flag = true;
9         int value;
10         if (flag)
11         {
12             Console.WriteLine($"Inside the code block: {value}");
13         }
14         value = 10;
15         Console.WriteLine($"Outside the code block: {value}");
16     }
17 }
```

Terminal Output:

```
Not alphanumeric character: ,
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9> dotnet run
Measured value is -4; too low.
Measured value is 5.
Measured value is 30; too high.
Failed measurement.
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9>
Historial restaurado
```

Compilation Error:

```
C:\Users\MI\PC\Documents\semestre-1\pc_semana9\Program.cs(8,18): error CS1002: Se esperaba ;
No se pudo llevar a cabo la compilación. Corrija los errores de compilación y vuelva a ejecutar el proyecto.
PS C:\Users\MI\PC\Documents\semestre-1\pc_semana9>
```

5.

```
// Code sample 1
bool flag = true;
int value;

if (flag)
{
    value = 10;
    Console.WriteLine($"Inside the code block: {value}");
}

Console.WriteLine($"Outside the code block: {value}");
```

```
c#

// Code sample 2
int value;

if (true)
{
    value = 10;
    Console.WriteLine($"Inside the code block: {value}");
}

Console.WriteLine($"Outside the code block: {value}");
```

6.

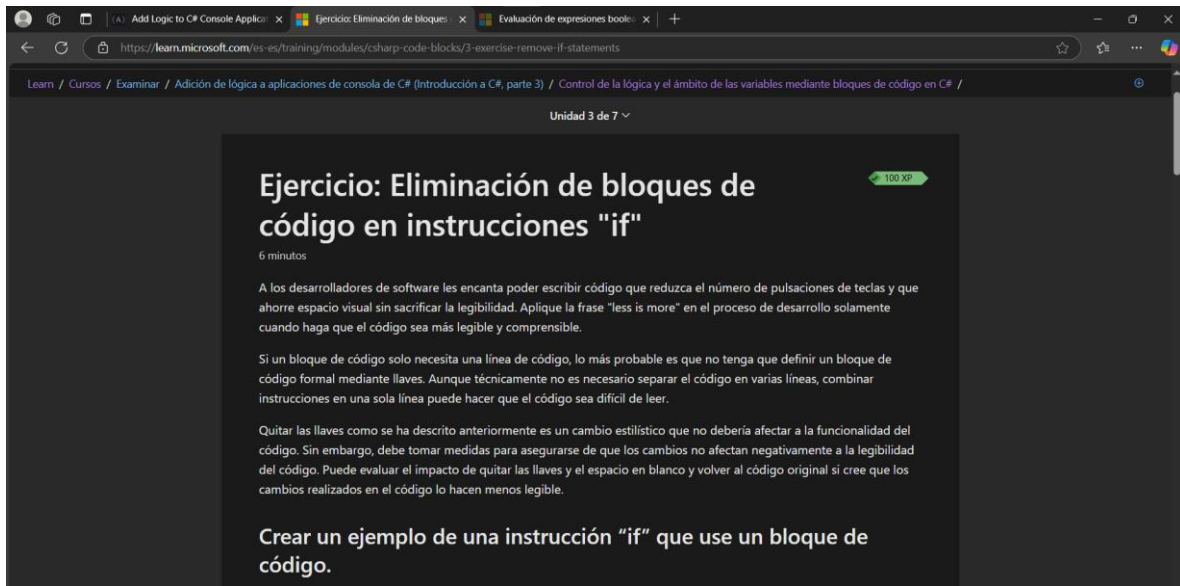
Unidad 2 de 7

Comprobación de conocimientos

1. Un desarrollador escribe código que incluye un bloque de código de la instrucción `if`. Inician una variable de entero en un valor de 5 por encima (fuera) del bloque de código. Inician una segunda variable de entero en un valor de 6 en la primera línea dentro del bloque de código. La expresión booleana del bloque de código de bloque se evalúa como `true` si la primera variable de entero tiene un valor superior a 0. En la segunda línea del bloque de código, asignan la suma de los dos valores a la primera variable. En la primera línea después del bloque de código, escriben código para mostrar el valor del primer entero. ¿Cuál es el resultado cuando se ejecuta la instrucción de código utilizada para mostrar el primer entero? *

- ☐ No se genera ningún error y se muestra el valor entero. El valor que se muestra es la suma del primer y segundo enteros.
- ☒ Correcto. Puesto que el primer entero se inicializa encima del código de instrucción `if`, sigue en el ámbito después del bloque de código. Además, dado que ambos enteros están en el ámbito y se inicializan con valores dentro del bloque de código, la adición de los valores se ejecuta correctamente. Por último, aunque el segundo entero no exista fuera del bloque de código, el primer entero conserva los cambios de su valor que se produjeron dentro del bloque de código.
- ☐ No se genera ningún error y se muestra el valor entero. El valor mostrado es el valor inicializado que figura encima del bloque de código.
- ☐ Se genera un error porque la primera variable no se encuentra en el ámbito después del bloque de código.

7.



The screenshot shows a web browser window with the URL <https://learn.microsoft.com/es-es/training/modules/csharp-code-blocks/3-exercise-remove-if-statements>. The page is titled "Ejercicio: Eliminación de bloques de código en instrucciones 'if'" and is part of "Unidad 3 de 7". It includes a progress indicator showing "100 XP" and a duration of "6 minutos". The text explains the importance of code readability and the "less is more" principle. It discusses how removing code blocks can improve readability and how removing braces can affect code readability. The exercise goal is to create an example of an 'if' statement using a code block.

Unidad 3 de 7

Ejercicio: Eliminación de bloques de código en instrucciones "if"

6 minutos

A los desarrolladores de software les encanta poder escribir código que reduzca el número de pulsaciones de teclas y que ahorre espacio visual sin sacrificar la legibilidad. Aplique la frase "less is more" en el proceso de desarrollo solamente cuando haga que el código sea más legible y comprensible.

Si un bloque de código solo necesita una línea de código, lo más probable es que no tenga que definir un bloque de código formal mediante llaves. Aunque técnicamente no es necesario separar el código en varias líneas, combinar instrucciones en una sola línea puede hacer que el código sea difícil de leer.

Quitar las llaves como se ha descrito anteriormente es un cambio estilístico que no debería afectar a la funcionalidad del código. Sin embargo, debe tomar medidas para asegurarse de que los cambios no afectan negativamente a la legibilidad del código. Puede evaluar el impacto de quitar las llaves y el espacio en blanco y volver al código original si cree que los cambios realizados en el código lo hacen menos legible.

Crear un ejemplo de una instrucción "if" que use un bloque de código.