

- Portada
- Índice
- Introducción
- Descripción
- Justificación
- Desarrollo
- Algoritmo
- Codificación
- Mejora del código
- Resultado en consola
- Conclusión
- Referencias bibliográficas



Actividad [#] -1

[Nombre de la Actividad] Mejora de la codificación

[Nombre del Curso] Seminario de titulación II

Ingeniería en Desarrollo de Software

Tutor: Elizabeth Guevara

Alumno: Daniel Alcudia Almeyda

Fecha: 18/11/2023

- Índice

Contenido

• Índice	3
• Introducción	3
• Descripción	4
• Justificación	4
• Desarrollo	5
• Algoritmo	5
• Codificación	5
• Mejora del código	6
• Resultado en consola	6
• Conclusión	7
• Referencias bibliográficas	8

- Introducción

La mejora en la codificación es fundamental en el mundo de la tecnología y la programación. Consiste en optimizar el proceso de escritura de código para hacerlo más eficiente, legible, mantenible y, en última instancia mejora el rendimiento del software desarrollado. Este proceso implica diversas estrategias y enfoques, desde la adopción de prácticas de codificación limpia y buenas

convenciones de estilo, hasta el uso de algoritmos más eficientes y técnicas de programación más avanzadas. La mejora en la codificación también abarca la optimización del código para reducir el consumo de recursos, como memoria y procesamiento, y para garantizar la seguridad y la escalabilidad del sistema.

La mejora de la codificación se refiere a la práctica de escribir código de manera más eficiente, legible y mantenible. Al utilizar técnicas de mejora de la codificación. Por otro lado el FizzBuzz es un programa clásico de la programación que se utiliza comúnmente en entrevistas técnicas para evaluar las habilidades de los candidatos.

- Descripción

La mejora en la codificación es un proceso continuo y multifacético que busca optimizar la calidad, eficiencia y mantenibilidad del código en desarrollo de software. Esto se logra a través de varias prácticas y enfoques:

Escribir código claro y comprensible facilita su lectura y comprensión por parte de otros desarrolladores, la mejora en la codificación también implica escribir código que funcione de manera óptima, minimizando la complejidad y maximizando el rendimiento, escribir código que sea fácil de mantener actualizar es crucial en el desarrollo a largo plazo, mejorar la codificación también involucra escribir código seguro para proteger contra vulnerabilidades posibles ataques. Validar entradas de usuario, evitar vulnerabilidades conocidas y seguir buenas prácticas de seguridad son aspectos importantes, la mejora en la purificación incluye la implementación de pruebas unitarias funcionales para garantizar que el código funcione correctamente, mantenerse al día con las nuevas tecnologías, herramientas y prácticas de desarrollo ayuda a mejorar constantemente la calidad del código y aprovechar las últimas innovaciones.

- Justificación

Mejorar la codificación implica encontrar formas más eficientes de resolver el problema. En el caso del FizzBuzz, cuál es más eficiente puede ejecutarse más rápido como consumir menos recursos y ser escalable, lo que es crucial en aplicaciones o sistemas con grandes cantidades de datos. Mejorar en la codificación y abordar el FizzBuzz de manera más eficiente promueve buenas prácticas de ingeniería. Esto incluye el uso adecuado de algoritmos, optimización de recursos y el desarrollo de habilidades clave para enfrentar problemas más complejos en el desarrollo de software.

La búsqueda de mejoras que notificación y enfoques para el FizzBuzz es una excelente oportunidad para aprender y crecer como desarrollador. Explorar

diferentes soluciones fomenta el pensamiento crítico y la creatividad para resolver problemas, habilidades cruciales en el campo de la programación.

Emplear mejoras en la codificación o solo optimiza la resolución de un problema aparentemente simple, sino que también promueve un enfoque más sólido y eficiente en el desarrollo de software, impulsando la calidad, la comprensión y la capacidad de adaptación del código en diferentes contextos y situaciones.

- Desarrollo
- Algoritmo

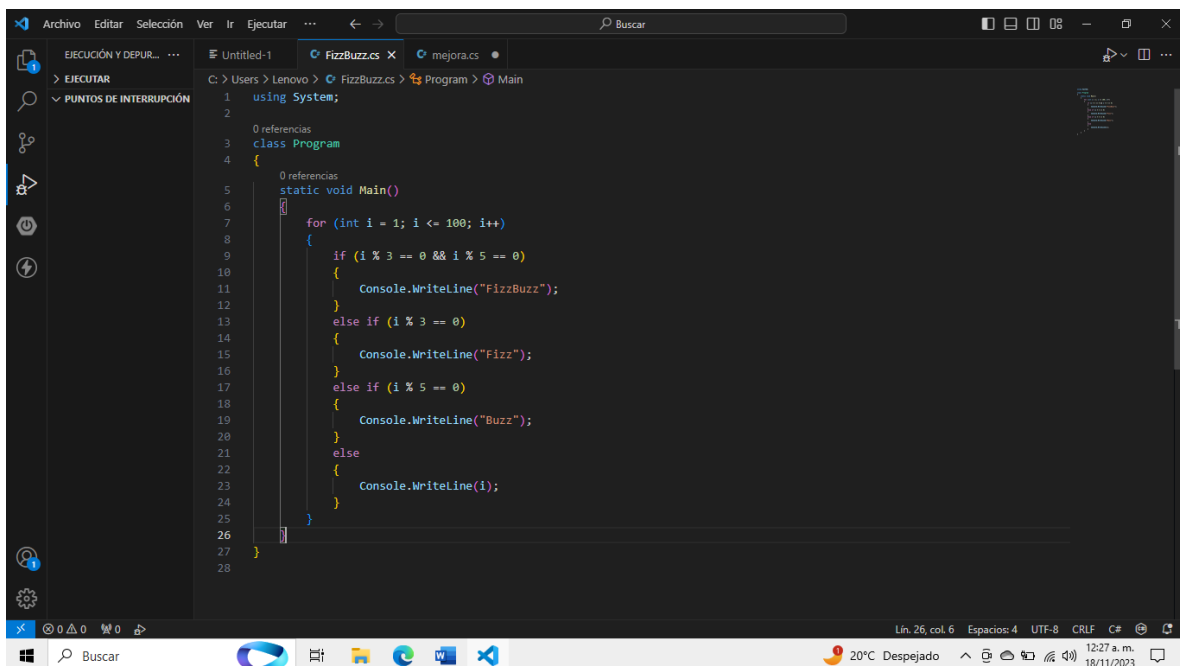
Si el número es divisible por 3, se imprime Fizz en lugar del número.

Si el número es divisible por 5, se imprime Buzz en lugar del número.

Si el número es divisible por los dos 3 y 5, se imprime FizzBuzz en lugar del número.

Si no se cumple ninguna condición se imprime el número.

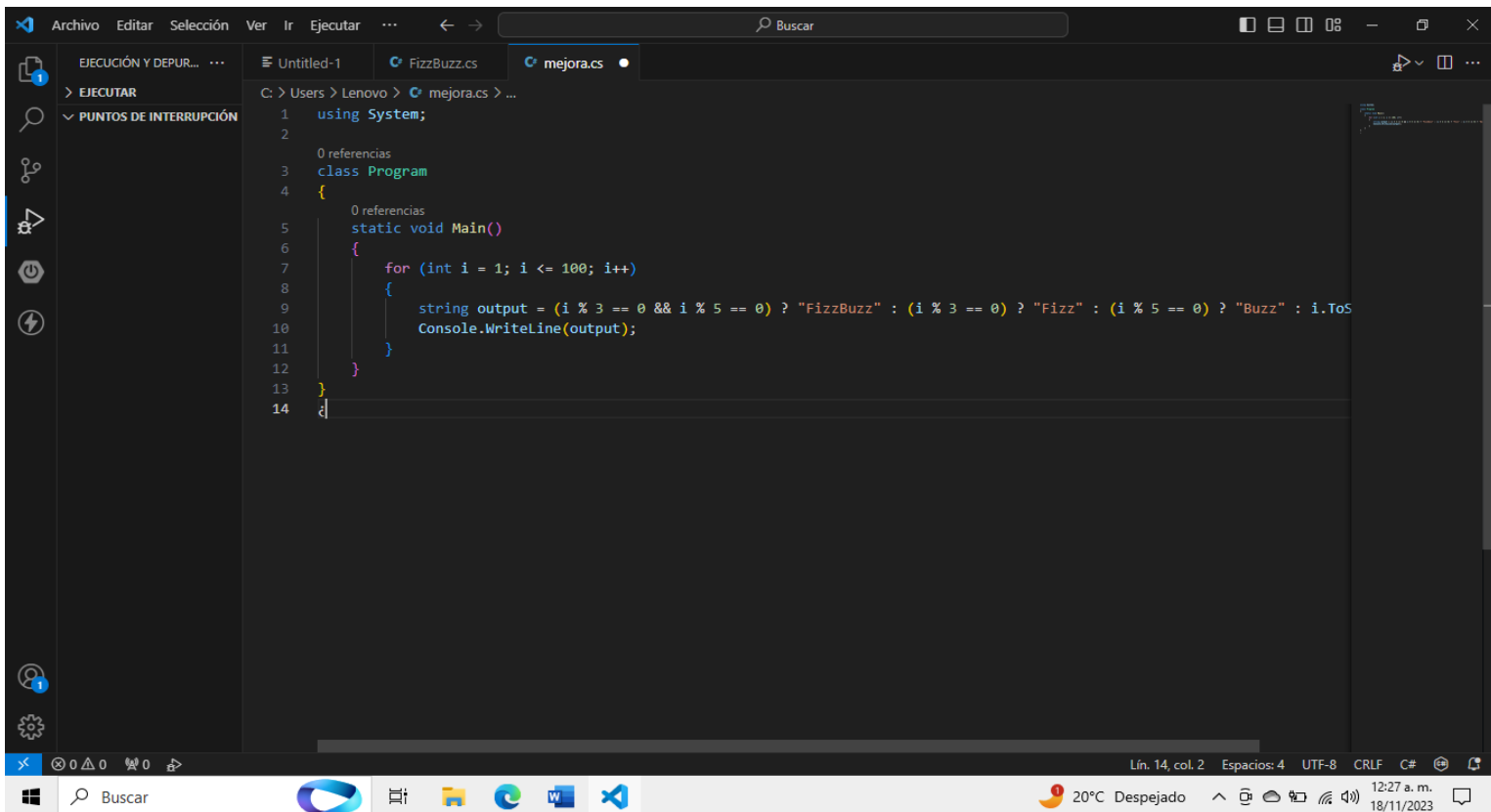
- Codificación



```
1  using System;
2
3  0 referencias
4  class Program
5  {
6      0 referencias
7      static void Main()
8      {
9          for (int i = 1; i <= 100; i++)
10         {
11             if (i % 3 == 0 && i % 5 == 0)
12             {
13                 Console.WriteLine("FizzBuzz");
14             }
15             else if (i % 3 == 0)
16             {
17                 Console.WriteLine("Fizz");
18             }
19             else if (i % 5 == 0)
20             {
21                 Console.WriteLine("Buzz");
22             }
23             else
24             {
25                 Console.WriteLine(i);
26             }
27         }
28     }
29 }
```

The screenshot shows the Visual Studio Code interface with a C# file named 'FizzBuzz.cs'. The code implements the FizzBuzz algorithm, iterating from 1 to 100. It checks if the current number is divisible by both 3 and 5 (FizzBuzz), then by 3 (Fizz), then by 5 (Buzz), and finally prints the number itself if none of the conditions are met. The interface includes a menu bar (Archivo, Editar, Selección, Ver, Ir, Ejecutar), a search bar, and a sidebar with icons for Explorer, Search, and Run and Debug. The status bar at the bottom shows 'Lín. 26, col. 6', 'Espacios: 4', 'UTF-8', 'CRLF', 'C#', and the system clock '12:27 a.m. 18/11/2023'.

En esta actividad se creó un programa que nos imprime los números del 1 al 100. Se utilizó la estructura for para poder iterar los números del 1 al 100, se utilizó la condición if, else para saber si los números son divisibles y nos pueda imprimir fizz, buzz, FizzBuzz, se utilizaron operadores. Se utilizó el método Console.WriteLine para poder imprimir el resultado de lo solicitado en la actividad.

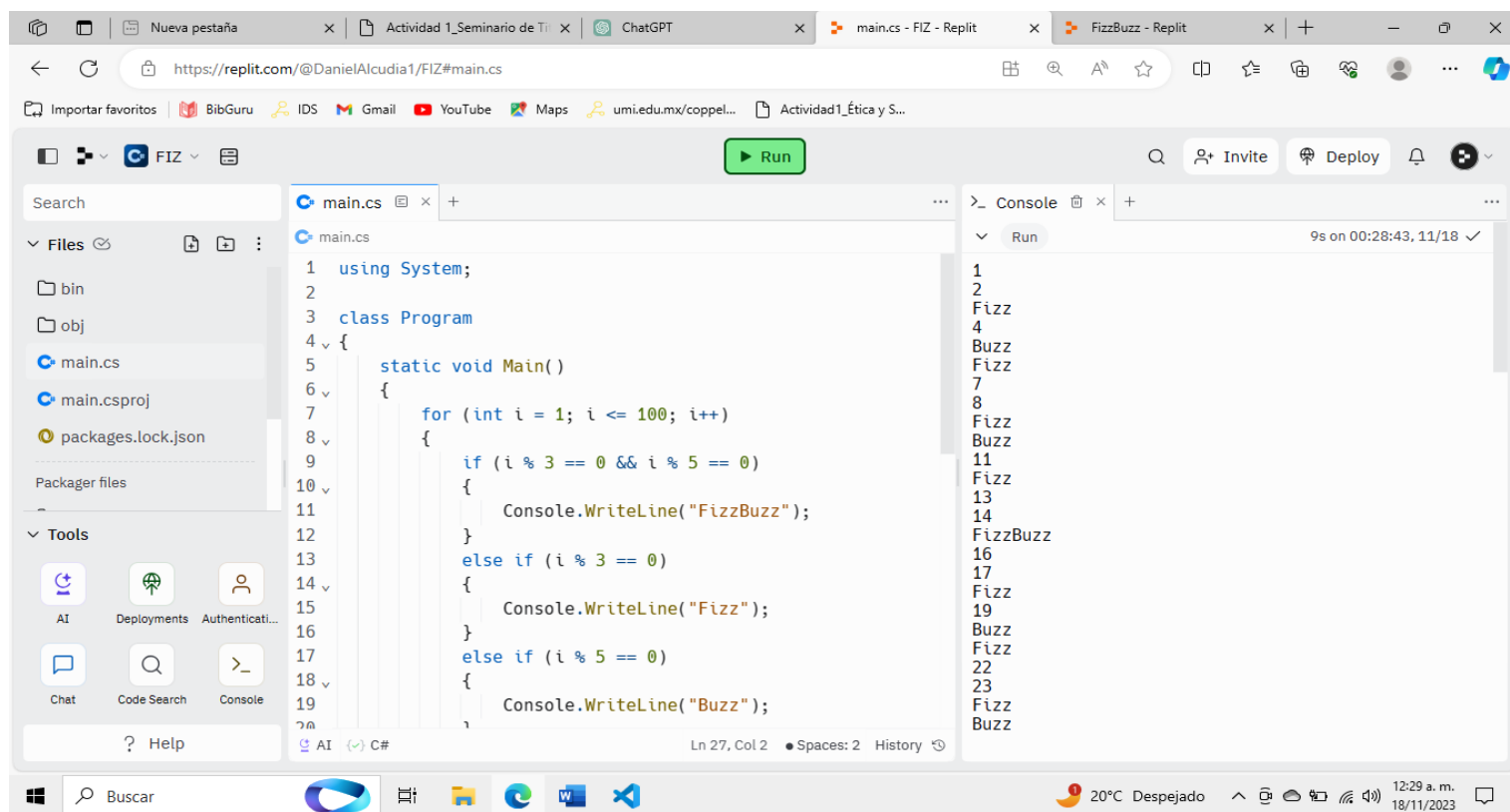
A screenshot of the Visual Studio Code editor interface. The top menu bar includes 'Archivo', 'Editar', 'Selección', 'Ver', 'Ir', 'Ejecutar', and a search bar. The left sidebar shows the 'Ejecución y Depuración' (Execution and Debugging) panel with 'Puntos de Interrupción' (Breakpoints) and a 'Run and Debug' button. The main editor area displays a C# file named 'mejora.cs' with the following code:

```
1 using System;
2
3 0 referencias
4 class Program
5 {
6     0 referencias
7     static void Main()
8     {
9         for (int i = 1; i <= 100; i++)
10        {
11            string output = (i % 3 == 0 && i % 5 == 0) ? "FizzBuzz" : (i % 3 == 0) ? "Fizz" : (i % 5 == 0) ? "Buzz" : i.ToString();
12            Console.WriteLine(output);
13        }
14    }
15 }
```

The bottom status bar shows 'Lín. 14, col. 2', 'Espacios: 4', 'UTF-8', 'CRLF', 'C#', and a system tray with a temperature indicator (20°C), a weather icon (Despejado), and the date/time (12:27 a.m., 18/11/2023).

Las mejoras que se le realizaron al código fue en la optimización de la impresión del resultado utilizando solo la instrucción Console.WriteLine para poder mostrar fizz, buzz.

• Resultado en consola

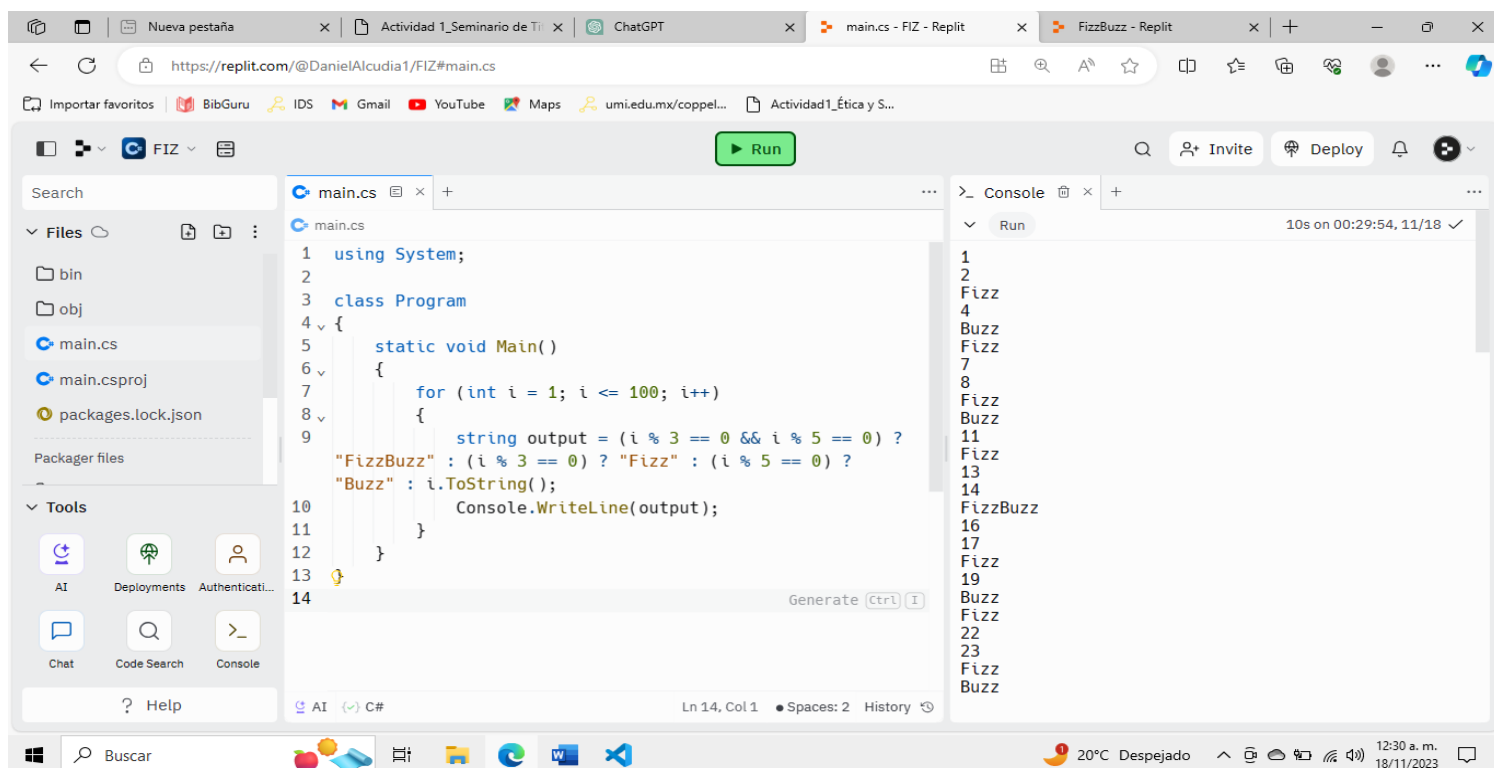


The screenshot shows a Replit IDE window with the URL `https://replit.com/@DanielAlcudia1/FIZ#main.cs`. The code in `main.cs` is as follows:

```
1 using System;
2
3 class Program
4 {
5     static void Main()
6     {
7         for (int i = 1; i <= 100; i++)
8         {
9             if (i % 3 == 0 && i % 5 == 0)
10            {
11                Console.WriteLine("FizzBuzz");
12            }
13            else if (i % 3 == 0)
14            {
15                Console.WriteLine("Fizz");
16            }
17            else if (i % 5 == 0)
18            {
19                Console.WriteLine("Buzz");
20            }
21        }
22    }
23 }
```

The console output shows the results of the program execution:

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
Buzz
Fizz
22
23
Fizz
Buzz
```



The screenshot shows a Replit IDE window with the URL `https://replit.com/@DanielAlcudia1/FIZ#main.cs`. The code in `main.cs` is as follows:

```
1 using System;
2
3 class Program
4 {
5     static void Main()
6     {
7         for (int i = 1; i <= 100; i++)
8         {
9             string output = (i % 3 == 0 && i % 5 == 0) ?
10            "FizzBuzz" : (i % 3 == 0) ? "Fizz" : (i % 5 == 0) ?
11            "Buzz" : i.ToString();
12            Console.WriteLine(output);
13        }
14    }
15 }
```

The console output shows the results of the program execution:

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
Buzz
Fizz
22
23
Fizz
Buzz
```

● Conclusión

La mejora en la codificación es fundamental en mi campo laboral en la vida cotidiana, ya que impacta directamente en la eficiencia, calidad y confiabilidad del software que se desarrolla. En mi trabajo, la aplicación de prácticas para mejorar la codificación garantiza la creación de programas más estables, seguros y fáciles de mantener, lo que resulta en una mejor experiencia para los usuarios finales y en una mayor eficiencia en los procesos de desarrollo.

En la vida cotidiana, la comprensión de conceptos sobre mejora de la codificación me permite apreciar la importancia de la calidad del software en otros dispositivos y aplicaciones que utilizo diariamente. Esto me hace ser más consciente de la importancia de las actualizaciones, la seguridad y la estabilidad en el software que forma parte de mi rutina, incluyendo mi confianza y comodidad al interactuar con la tecnología. La mejora en la aplicación no sólo es esencial para impulsar la excelencia en el desarrollo del software, sino que también tiene un impacto directo en la calidad de vida y la eficiencia en el uso de la tecnología en mi campo laboral y en mi día a día.

● Referencias bibliográficas

(S/f). Openai.com. Recuperado el 19 de noviembre de 2023, de

<https://chat.openai.com/c/f6fea82b-ac3d-4d8d-bd8e-c49cc994f821>

(S/f-b). Openai.com. Recuperado el 19 de noviembre de 2023, de

<https://chat.openai.com/c/117a6e33-b30d-465d-ad52-5c67e98f66bo>