

Evaluación del 1er Parcial (Parte Práctica).

Aleman Montenegro Josue

Por favor todo el código compartido en Github, adjunte a las preguntas finales el repositorio con el código y compártelo en un documento PDF con su nombre, asignatura y el nombre del docente.

Ejercicio 1: Variables y Operadores en C#

1. Declarar dos variables, **numero1** y **numero2**, e inicialízalas con valores numéricos.
2. Calcula la suma de estas dos variables y almacena el resultado en una tercera variable llamada **resultado**.
3. Imprime en la consola el valor de resultado.

```
Microsoft Visual Studio Debug Console
La suma de 5 y 10 es: 15
C:\Users\eljos\Documents\ESPE 23-24\ADVANCED WEB\EXAMEN_ALEMAN_JOSUE\
ss 7720) exited with code 0.
Press any key to close this window . . .|
```

Ejercicio 2: Estructuras de Control en C#

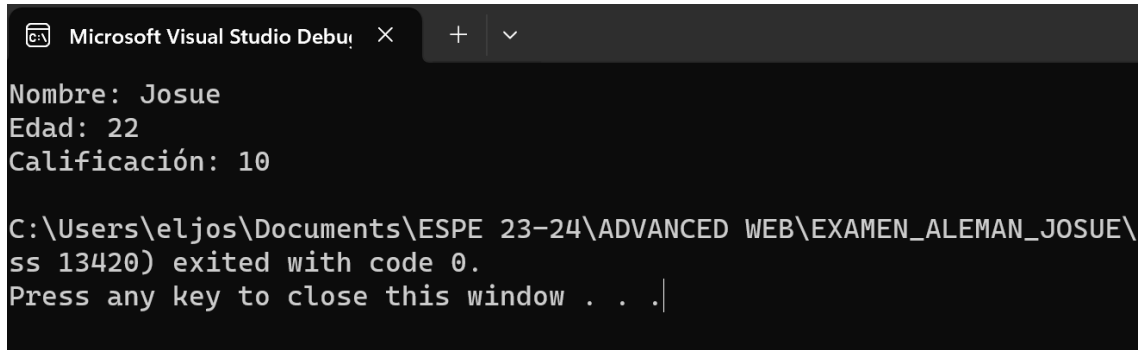
1. Declara una variable **edad** e inicialízala con un valor numérico.
2. Utiliza una estructura **if** para determinar si la persona es mayor de edad (mayor o igual a 18).
3. Imprime en la consola un mensaje indicando si la persona es mayor de edad o no.

```
Microsoft Visual Studio Debug Console
Ingresa tu edad: 21
La persona es mayor de edad.
C:\Users\eljos\Documents\ESPE 23-24\ADVANCED WEB\EXAMEN_ALEMAN_JOSUE\
ss 18772) exited with code 0.
Press any key to close this window . . .|
```

Ejercicio 3: Programación Orientada a Objetos - Clases y Objetos

1. Crea una clase llamada **Estudiante** con propiedades como **Nombre**, **Edad** y **Calificación**.
2. Crea un objeto de tipo **Estudiante** llamado **estudiante1** e inicializa sus propiedades con valores ficticios.

3. Imprime en la consola la información del estudiante.

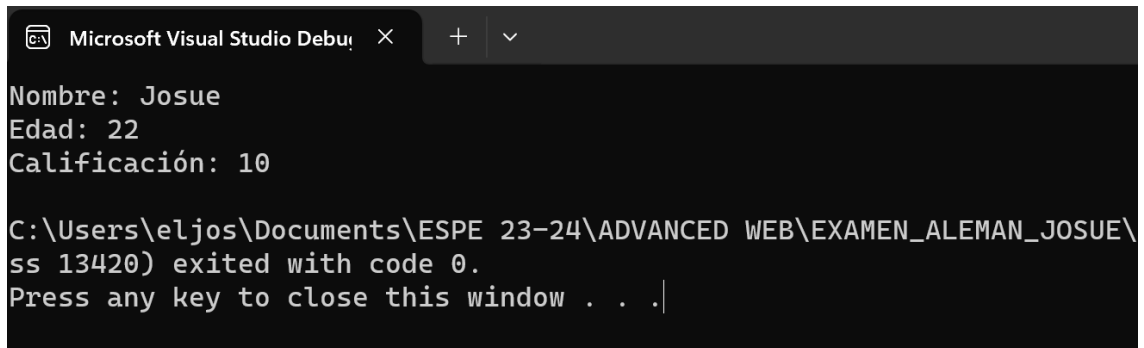


```
Microsoft Visual Studio Debug Console
Nombre: Josue
Edad: 22
Calificación: 10

C:\Users\eljos\Documents\ESPE 23-24\ADVANCED WEB\EXAMEN_ALEMAN_JOSUE\
ss 13420) exited with code 0.
Press any key to close this window . . .|
```

Ejercicio 4: Programación Orientada a Objetos - Métodos

1. Amplía la clase Estudiante con un método llamado **MostrarInformacion** que imprima en la consola los detalles del estudiante.
2. Llama al método **MostrarInformacion** para el objeto **estudiante1** y observa la salida.



```
Microsoft Visual Studio Debug Console
Nombre: Josue
Edad: 22
Calificación: 10

C:\Users\eljos\Documents\ESPE 23-24\ADVANCED WEB\EXAMEN_ALEMAN_JOSUE\
ss 13420) exited with code 0.
Press any key to close this window . . .|
```

Ejercicio 5: Programación Orientada a Objetos - Herencia

1. Crea una nueva clase llamada **EstudianteGraduado** que herede de la clase **Estudiante**.
2. Añade una nueva propiedad a **EstudianteGraduado** llamada **Titulo** que almacene el título obtenido.

3. Crea un objeto de tipo **EstudianteGraduado** llamado **graduado1** e inicializa sus propiedades.
4. Utiliza el método **MostrarInformacion** de la clase base para mostrar la información del estudiante graduado.

```
Microsoft Visual Studio Debug Console
Nombre: JuanPa
Edad: 24
Calificación: 10
Título obtenido: Ingeniero de Software
C:\Users\eljso\Documents\ESPE 23-24\ADVANCED WEB\EXAMEN_ALEMAN_JOSUE\
ss 29780) exited with code 0.
Press any key to close this window . . .|
```

Preguntas de Reflexión:

1. ¿Cuál es la diferencia entre una variable y una propiedad en C#?

Una variable es una ubicación de almacenamiento que contiene un valor, y puede ser una parte de una clase o un método. Una propiedad es una forma de exponer o controlar el acceso a los campos de una clase, generalmente con lógica adicional en los métodos get y set.

2. Explica cómo funciona la estructura **if** y por qué es útil en programación.

If ejecuta una acción en función de una condición.
Es útil para tomar decisiones en el funcionamiento de un sistema.

3. ¿Qué ventajas ofrece la programación orientada a objetos en comparación con otros paradigmas de programación?

La aplicación de los conceptos de la programación orientada a objetos, desde mi perspectiva:

- Facilita el mantenimiento de la aplicación.
- Optimiza recursos, encapsulando y abstrayendo información.
- Organiza los elementos del sistema mediante clases y módulos, facilitando la reutilización del código

4. ¿Cuándo usarías la herencia en un diseño de clases?

Cuando identifique que dos o más objetos comparten las mismas características de otra clase en particular.

5. ¿Por qué es importante la encapsulación en programación orientada a objetos?

Los detalles internos de una clase se mantienen ocultos, lo que facilita cambios sin afectar el resto del programa.

LINK DEL REPOSITORIO:

[josuealeman17/ALEMAN_EXAMEN \(github.com\)](https://github.com/josuealeman17/ALEMAN_EXAMEN)