“UNIVERSIDAD DE LAS FUERZAS ARMADAS” ESPE

INGENIERÍA DE SOFTWARE

DESARROLLO WEB AVANZADO

NOMBRES:

LISBETH CARVAJAL

TUTOR ENCARGADO:

ING.DIEGO SAAVEDRA

NRC:
14956

FECHA DE ENTREGA:
13 de diciembre de 2023

SEDE MATRIZ SANGOLQUÍ
QUITO-ECUADOR 2023

Evaluación del 1er Parcial (Parte Práctica).

Por favor todo el código compartido en Github, adjunte a las preguntas finales el repositorio
con el código y compartelo 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.

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.

Ejercicio 3: Programación Orientada a Objetos - Clases y Objetos 1. Crea una clase llamada Estudiante con propiedades como Nombre, Edad y Calificacion.

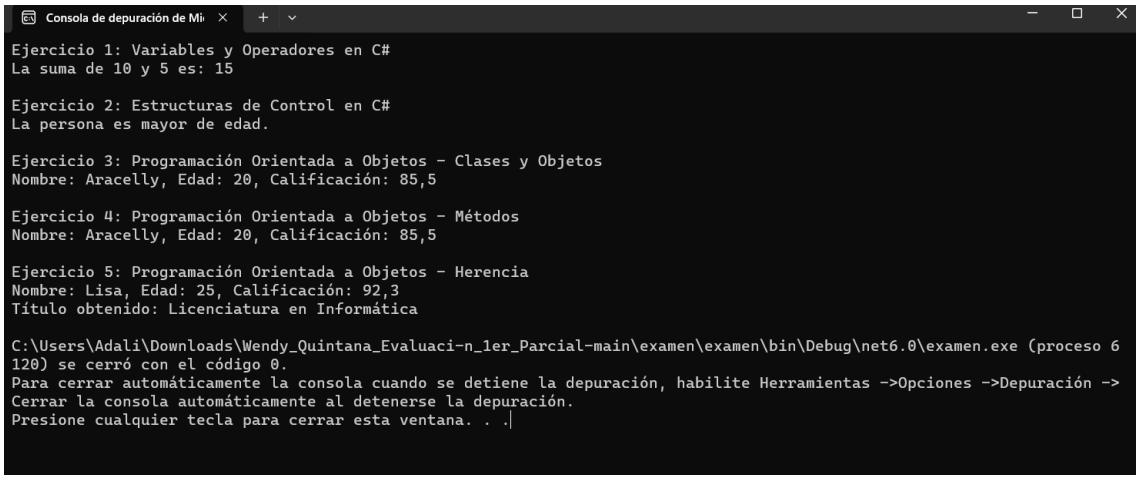
2. Crea un objeto de tipo Estudiante llamado estudiante1 e inicializa sus propiedades con valores ficticios.
3. Imprime en la consola la información del estudiante.

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.

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.



Preguntas de Reflexión:

1. Diferencia entre variable y propiedad en C#:

- Una variable es un espacio de almacenamiento nombrado que puede contener un valor o una referencia a un objeto. Se declara utilizando un tipo de datos y puede cambiar durante la ejecución del programa.
- Una propiedad, por otro lado, es una abstracción de los accesos de lectura y escritura a un campo de clase. Las propiedades generalmente tienen un método "get" para obtener el valor y un método "set" para asignar un valor. Proporcionan un nivel de encapsulación y control sobre el acceso a los datos.

2. Estructura if y su utilidad:

- La estructura "if" es una construcción de control de flujo en programación que permite ejecutar un bloque de código si una condición especificada es verdadera.
- Funciona evaluando la condición booleana. Si la condición es verdadera, el bloque de código dentro del "if" se ejecuta; de lo contrario, se salta ese bloque.
- Es útil porque permite que un programa tome decisiones basadas en condiciones. Puedes realizar diferentes acciones según si una condición es verdadera o falsa, lo que brinda flexibilidad y control en la ejecución del programa.

3. Ventajas de la programación orientada a objetos (POO):

- Reutilización de código: La POO permite la creación de clases y objetos que pueden ser reutilizados en diferentes partes del programa, lo que ahorra tiempo y esfuerzo.
- Modularidad: La modularidad se logra mediante la encapsulación y la abstracción, lo que facilita la gestión y mantenimiento del código.
- Facilita el modelado del mundo real: Los objetos en la POO representan entidades del mundo real y sus interacciones, lo que facilita la comprensión y el diseño de sistemas.

- Herencia y polimorfismo: Estos conceptos permiten la creación de jerarquías de clases y la implementación de interfaces, lo que mejora la estructura y extensibilidad del código.

4. Uso de la herencia en un diseño de clases:

- La herencia se utiliza cuando se quiere crear una nueva clase que comparta propiedades y comportamientos de una clase existente (clase base o padre). - Permite la reutilización de código y la extensión de funcionalidades, ya que la clase derivada (hija) hereda características de la clase base.
- Se utiliza cuando hay una relación de "es un" entre la clase base y la clase derivada. Por ejemplo, si tienes una clase "Animal", podrías tener clases derivadas como "Perro" y "Gato".

5. Importancia de la encapsulación en la POO:

- La encapsulación es el concepto de ocultar los detalles internos de una clase y proporcionar una interfaz pública para interactuar con ella.
- Ayuda a proteger los datos de la clase al limitar el acceso y la modificación directa desde fuera de la clase.
- Facilita el cambio interno de implementación sin afectar a las otras partes del programa que utilizan la clase.
- Contribuye a la modularidad y mantenimiento del código, ya que los cambios internos no afectan a otros componentes si la interfaz pública se mantiene constante.

Link del Repositorio:
https://github.com/adalicarvajal/Carvajal_Lisbeth_Evaluacion_1Parcial.git