FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

GUÍA DOCUMENTADA - INFORME DE PROYECTO

Autores:

- José luiz Garcia Moreno
- Sebastian Enrique Rodríguez Leiva
- José Carlos Jauregui Zavaleta
- Milagros Yeraldin Polo Niquin
- Frank Junior Alberto Flores Castillo
- Fernando Giuseppen Muñoz Asuncion

Curso: Técnicas de Programación Orientada a Objetos
Docente del Curso: Ing. Jorge Ricardo Perez Vigil

Trujillo – Perú
2025

🧩 TRABAJO DE CAMPO - S05: SOBRECARGA, MANEJO DE ERRORES Y COLECCIONES CON GIT

🔧 Desarrollo con Código

✅ 1. Sobrecarga de Métodos

```
public class Producto
{
    public double CalcularPrecio(int cantidad)
    {
        return cantidad * 10.0;
    }

    public double CalcularPrecio(int cantidad, double descuento)
    {
        return cantidad * 10.0 * (1 - descuento);
    }

    public double CalcularPrecio(string codigo)
    {
        if (codigo == "VIP") return 5.0;
        return 10.0;
    }
}
```

📌 Git aplicado:

```
git checkout -b feature/sobrecarga
git add Producto.cs
git commit -m "Implementa sobrecarga de método CalcularPrecio"
```

✅ 2. Manejo de Errores

```
try
{
    Console.Write("Ingrese número: ");
    int numero = int.Parse(Console.ReadLine());
    Console.WriteLine($"Número ingresado: {numero}");
}
catch (FormatException ex)
{
    Console.WriteLine("Error: Entrada no válida.");
}
catch (Exception ex)
{
    Console.WriteLine($"Error inesperado: {ex.Message}");
}
```

📌 Git aplicado:

```
git checkout -b feature/errores
git add EntradaUsuario.cs
git commit -m "Manejo de excepciones para entrada de datos"
```

✅ 3. Colecciones

```
List<string> tareas = new List<string>();
tareas.Add("Revisar código");
tareas.Add("Subir al repositorio");

foreach (string tarea in tareas)
{
    Console.WriteLine($"- {tarea}");
}
```

📌 Git aplicado:

```
git checkout -b feature/colecciones
git add Tareas.cs
git commit -m "Implementa manejo de tareas con List<string>"
```

✅ 4. Control en Repositorio GitHub

- Se creó un repositorio en GitHub llamado `gestion-tareas`.
- Se dividió el trabajo por ramas: `feature/sobrecarga`, `feature/errores`, `feature/colecciones`.
- Cada integrante hizo `commit`, `push` y `pull request`.
- El líder revisó los PR antes de fusionarlos a `main`.
