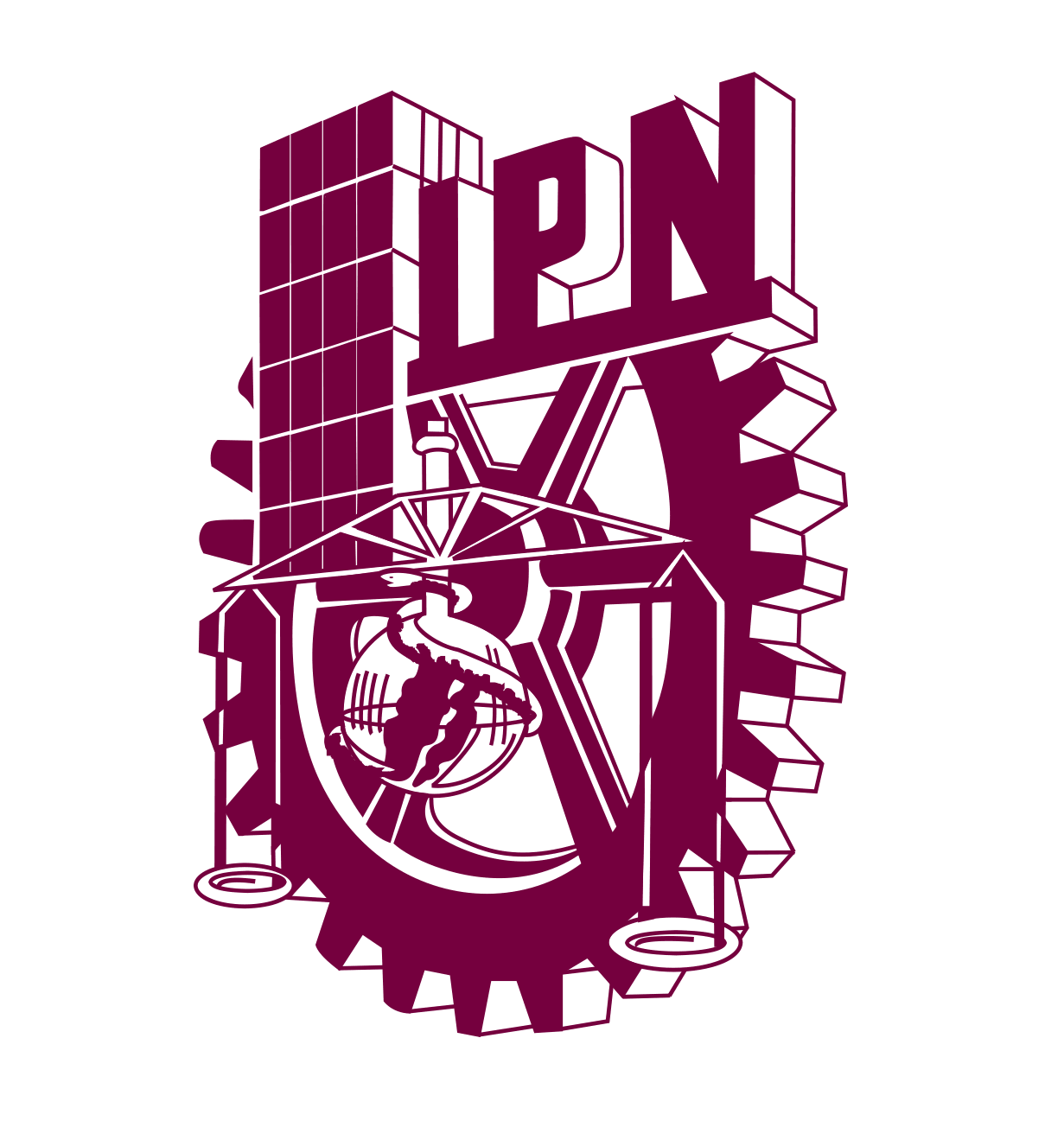
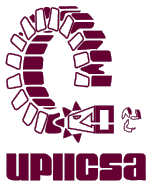
**INSTITUTO POLITÉCNICO NACIONAL**

**Unidad Profesional Interdisciplinaria de Ingeniería y Ciencias Sociales y Administrativas**

**Ingeniería de Pruebas**

**Documento Técnico: Diseño - Calculadora en Python con Tkinter**

**Autores:**

* Contreras Mosco Cristobal
* Díaz Pérez Diego
* Hernández Aguirre Ricardo
* Monroy Muñoz Angel Yael
* Salazar Rocha Any Jennifer

**Fecha:** [09/09/2025]

**Versión:** 1.0

**Índice**

1. Introducción
2. Objetivo
3. Arquitectura del sistema
4. Diseño de módulos
5. Diagramas

5.1 Diagrama de flujo

5.2 Diagrama de clases / módulos 5.3 Diseño de interfaz gráfica

1. Pseudocódigo
2. Estructuras de datos
3. Conclusión

**Introducción**

El presente documento describe el diseño de la calculadora desarrollada en Python con Tkinter, a partir de los requerimientos funcionales y no funcionales ya definidos. Este diseño establece la base técnica para la implementación, pruebas y mantenimiento del sistema, mediante la descripción de su arquitectura, módulos principales, diagramas representativos, algoritmos y estructuras de datos.

**Objetivo del diseño**

El objetivo del diseño es transformar los requerimientos en especificaciones técnicas claras. Con esto se busca:

* Definir una arquitectura modular que permita organización y mantenibilidad.
* Describir cómo se implementan las funciones solicitadas (operaciones, historial, validaciones).
* Mostrar visualmente el flujo del sistema y la interfaz.
* Facilitar la comprensión de los algoritmos clave mediante pseudocódigo.
* Establecer una base para las futuras pruebas y posibles mejoras.

**Arquitectura del sistema**

El sistema se divide en cuatro bloques principales:

| Interfaz gráfica (UI): | Lógica de operaciones: | Validaciones y manejo de errores: | Historial de operaciones: |
| --- | --- | --- | --- |
| Construida en Tkinter.  Incluye pantalla, botones (números, operadores, comandos) e historial.  Es el puente entre el usuario y la lógica de operaciones. | Procesa sumas, restas, multiplicaciones y divisiones.  Incluye redondeo automático a 2 decimales y validación de resultados enteros. | Detección de división entre cero.  Prevención de operadores consecutivos.  Restricción de un único punto decimal por número.  Límite de 12 caracteres en pantalla. | Registra dinámicamente cada operación.  Puede mostrarse, ocultarse o borrarse.  Se gestiona en tiempo de ejecución, no se guarda tras cerrar la aplicación. |

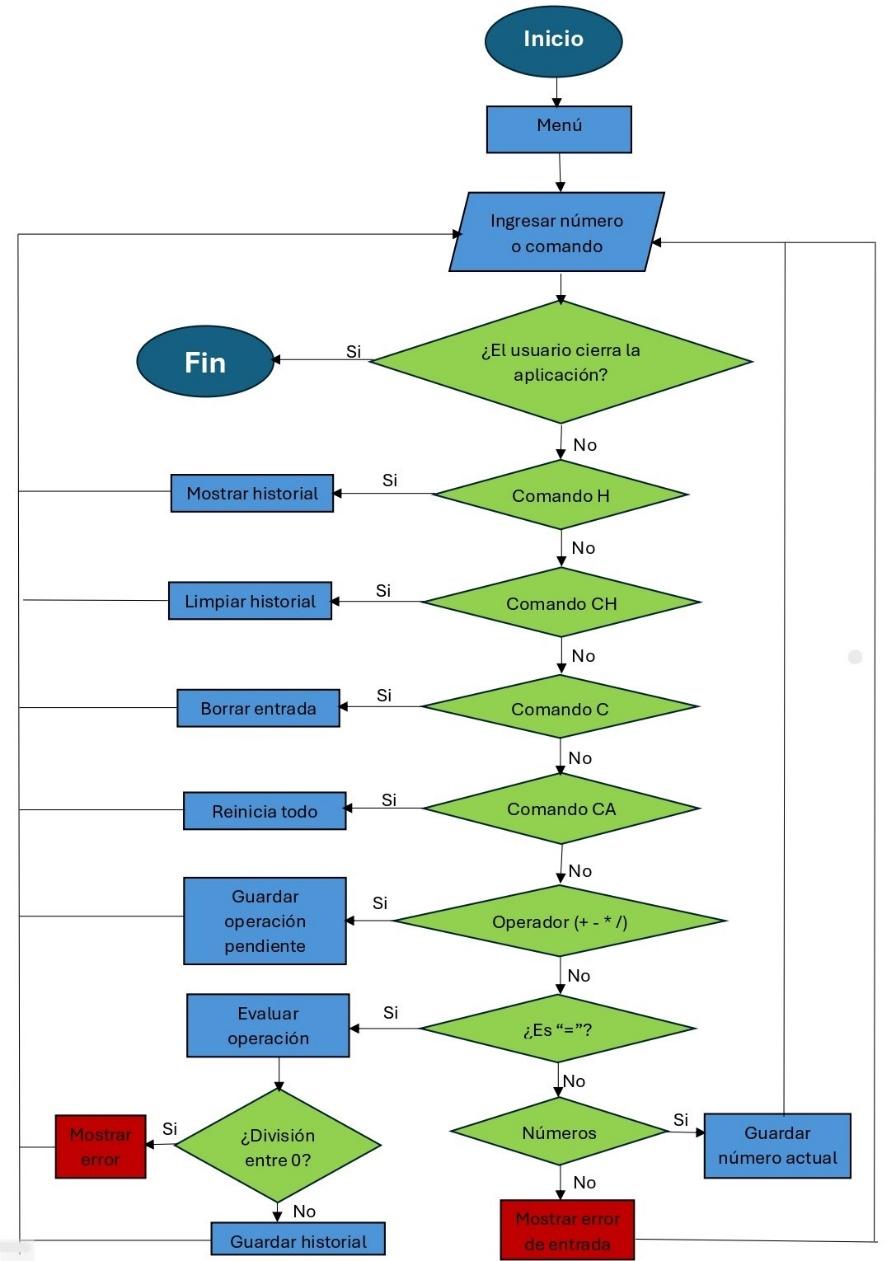
**Diseño de módulos**

**Atributos:**

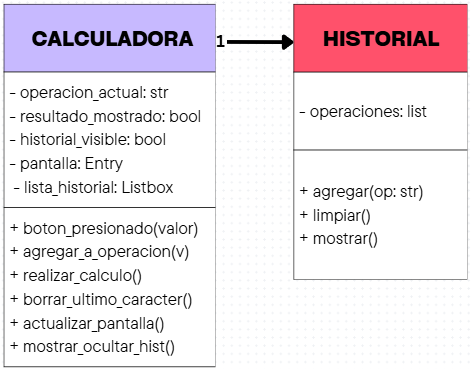
* operacion\_actual: cadena que guarda la operación en curso.
* resultado\_mostrado: booleano que indica si la pantalla muestra un resultado.
* historial\_visible: booleano que controla si el historial se muestra.

**Métodos:**

* boton\_presionado(valor): gestiona la acción según el botón pulsado.
* agregar\_a\_operacion(valor): construye las expresiones matemáticas.
* realizar\_calculo(): evalúa la operación y maneja errores.
* borrar\_ultimo\_caracter(): elimina el último carácter de la operación.
* actualizar\_pantalla(): refresca el contenido de la pantalla.
* mostrar\_ocultar\_historial(): alterna entre mostrar y ocultar el historial.

**Diagramas**

**Diagrama de flujo**

**Diagrama de clases / módulos (UML)**

**Pseudocódigo**

**realizar\_calculo()**

INICIO

SI operacion\_actual está vacía O termina en operador

MOSTRAR "ERROR: Incompleta"

operacion\_actual = ""

SINO

INTENTAR evaluar operacion\_actual

SI resultado es decimal

REDONDEAR a 2 decimales

SI resultado es entero

MOSTRAR sin punto decimal

GUARDAR operación en historial

MOSTRAR resultado en pantalla

operacion\_actual = resultado

resultado\_mostrado = VERDADERO

CAPTURAR ZeroDivisionError

MOSTRAR "ERROR: ÷0"

operacion\_actual = ""

CAPTURAR Exception

MOSTRAR "ERROR"

operacion\_actual = ""

FIN

**agregar\_a\_operacion(valor)**

INICIO

SI resultado\_mostrado ES VERDADERO

SI valor ES operador

operacion\_actual = pantalla + valor

SINO

operacion\_actual = valor

resultado\_mostrado = FALSO

SINO

SI longitud de operacion\_actual >= 12

RETORNAR

SI valor ES "."

VERIFICAR último número

SI ya contiene ".", RETORNAR

SI valor ES operador

SI último carácter también ES operador

REEMPLAZAR por nuevo operador

ACTUALIZAR pantalla

RETORNAR

CONCATENAR valor a operacion\_actual

ACTUALIZAR pantalla

FIN

**mostrar\_ocultar\_historial()**

INICIO

SI historial\_visible = VERDADERO

OCULTAR panel historial

historial\_visible = FALSO

SINO

MOSTRAR panel historial

historial\_visible = VERDADERO

FIN

**Estructuras de datos**

* Cadenas (str): para las expresiones (operacion\_actual).
* Booleanos (bool): para estados (resultado\_mostrado, historial\_visible).
* Listbox de Tkinter: lista dinámica para el historial.

**Conclusiones**

El diseño propuesto detalla la arquitectura modular, algoritmos, estructuras de datos y diagramas del sistema calculadora, lo cual asegura que el software cumpla con los requerimientos funcionales y no funcionales. La inclusión de pseudocódigo y pruebas preliminares facilita la validación del sistema y la futura extensibilidad hacia nuevas funciones (científicas, conversores, etc.).