

Manual de Python

- Semana 7:

Calculadora

Tkinter Parte 1

7.1. Introducción a Tkinter

Tkinter es una biblioteca estándar de Python que permite crear interfaces gráficas de usuario (GUI). Es fácil de usar y ofrece los componentes básicos necesarios para crear aplicaciones interactivas, como botones, cuadros de texto y ventanas.

En esta semana, implementaremos una **calculadora** utilizando Tkinter, donde los estudiantes aprenderán sobre interfaces gráficas y cómo organizar el código usando la **Programación Orientada a Objetos (POO)**.

7.2. Estructura del Código Original

Este archivo define una **calculadora gráfica** utilizando la biblioteca **Tkinter** para la creación de la interfaz. A continuación, explicaré cada sección del código.



Importación de Tkinter

```
from tkinter import *
```

Tkinter es la biblioteca estándar en Python para crear interfaces gráficas de usuario (GUIs). El `*` significa que se están importando todos los módulos y funciones de la biblioteca.

Creación de la Ventana Principal

```
ventana = Tk()
ventana.title("Calculadora")
```

Aquí, la variable `ventana` es una instancia de la clase `Tk()`, que es la base de cualquier aplicación en Tkinter. El método `title()` cambia el título de la ventana a "Calculadora".

Entrada de Texto

```
i = 0

e_texto = Entry(ventana, font=("Calibri 20"))
e_texto.grid(row=0, column=0, columnspan=4, padx=5, pady=5)
```

- `= 0`: Se utiliza para mantener la posición del texto en el cuadro de entrada.
- **Entry**: Se usa para crear un cuadro de texto donde se mostrará la operación y el resultado.
- **grid()**: Posiciona el cuadro de texto en la interfaz en la fila 0 y columna 0, ocupando 4 columnas (con `columnspan=4`).

Funciones de Operaciones

El código incluye tres funciones que realizan las operaciones principales:



Función click_boton(valor)

```
def click_boton(valor):  
    global i  
    e_texto.insert(i, valor)  
    i = i + 1
```

- **click_boton:** Inserta el valor (número u operador) en la posición *i* del cuadro de texto *e_texto*. Cada vez que se presiona un botón, *i* aumenta para que el siguiente carácter se inserte en la posición correcta.

Función borrar()

```
def borrar():  
    e_texto.delete(0, END)  
    i = 0
```

- **borrar:** Elimina todo el contenido del cuadro de texto. `delete(0, END)` borra desde el primer carácter (0) hasta el final del cuadro (END).

Función hacer_operacion()

```
def hacer_operacion():  
    ecuacion = e_texto.get()  
    resultado = eval(ecuacion)  
    e_texto.delete(0, END)  
    e_texto.insert(0, resultado)  
    i = 0
```

- **hacer_operacion:** Esta función toma la ecuación escrita en el cuadro de texto (`get()`), la evalúa con la función `eval()` y muestra el resultado en el cuadro. `eval()` es una función poderosa de Python que ejecuta una cadena como si fuera código Python.



Creación de Botones

Los botones de la calculadora se crean individualmente utilizando el widget **Button** de Tkinter.

Botones Numéricos y de Operadores

```
button1 = Button(ventana, text='1', width="5", height="2", command=
lambda: click_boton(1))
# Botones 2 a 9 y 0 son creados de la misma manera...
boton_sum = Button(ventana, text='+', width="5", height="2", command=
lambda: click_boton("+"))
boton_igual = Button(ventana, text='=', width="5", height="2", command=
lambda: hacer_operacion())
```

- **Button:** Cada botón se crea con el texto que se mostrará (`text`), su tamaño (`width`, `height`) y la función que ejecutará cuando se presione (`command`).
- Se usan **lambda** funciones para pasar los valores correspondientes a `click_boton()` o para realizar una operación con `hacer_operacion()`.

Botón de Borrar y Paréntesis

```
boton_borrar = Button(ventana, text='AC', width="5", height="2",
command=lambda: borrar())
boton_parentesis1 = Button(ventana, text='(', width="5", height="2",
command=lambda: click_boton("("))
boton_parentesis2 = Button(ventana, text=')', width="5", height="2",
command=lambda: click_boton("))"))
```

Estos botones tienen funcionalidades específicas como borrar todo el contenido (`borrar()`) o insertar paréntesis.

Posicionamiento de los Botones

```
boton_borrar.grid(row=1, column=0, padx=5, pady=5)
button7.grid(row=2, column=0, padx=5, pady=5)
# Otros botones se posicionan de manera similar
```



- `grid()`: Coloca cada botón en una celda específica de la cuadrícula, definida por las filas (`row`) y columnas (`column`).
- `padx` y `pady`: Añaden espacio de separación horizontal (x) y vertical (y) alrededor de los botones.

Bucle Principal de Tkinter



Este es el **bucle principal** de la interfaz gráfica. Es el que mantiene la ventana abierta y responde a los eventos de los botones y entradas de texto.

Manual de Python

- Semana 7.1.1:

Calculadora

Tkinter Parte 2

7.1.1 Clases y Objetos en Tkinter

Clase Calculadora



En la POO, una **clase** es un plano o plantilla para crear objetos. En este proyecto, la clase Calculadora representará nuestra interfaz gráfica, con todos los botones y funcionalidad de una calculadora básica.

Definiendo la clase Calculadora

```
from tkinter import *

class Calculadora:
    def __init__(self, ventana):
        self.ventana = ventana
        self.ventana.title("Calculadora")
        self.i = 0 # Contador para la posición del texto

        # Entrada de texto donde se mostrará la operación
        self.e_texto = Entry(self.ventana, font=("Calibri 20"))
        self.e_texto.grid(row=0, column=0, columnspan=4, padx=5, pady=5)

        # Inicialización de botones
        self.crear_botones()

    def click_boton(self, valor):
        self.e_texto.insert(self.i, valor)
        self.i += 1

    def borrar(self):
        self.e_texto.delete(0, END)
        self.i = 0

    def hacer_operacion(self):
        ecuacion = self.e_texto.get()
        try:
            resultado = eval(ecuacion) # Evalúa la operación
            self.e_texto.delete(0, END)
            self.e_texto.insert(0, resultado)
        except:
            self.e_texto.delete(0, END)
            self.e_texto.insert(0, "Error")
            self.i = 0

    def crear_botones(self):
        # Definir botones numéricos y operadores
        botones = [
            ('1', 4, 0), ('2', 4, 1), ('3', 4, 2), ('4', 3, 0), ('5', 3,
1), ('6', 3, 2),
```



```
1),  
    ('+', 3, 3), ('-', 4, 3), ('*', 2, 3), ('/', 1, 3),  
    ('(', 1, 1), (')', 1, 2), ('AC', 1, 0), ('=', 5, 2)  
]  
  
for (texto, fila, col) in botones:  
    if texto == 'AC':  
        boton = Button(self.ventana, text=texto, width=5,  
height=2, command=self.borrar)  
    elif texto == '=':  
        boton = Button(self.ventana, text=texto, width=5,  
height=2, command=self.hacer_operacion)  
    else:  
        boton = Button(self.ventana, text=texto, width=5,  
height=2, command=lambda t=texto: self.click_boton(t))  
  
    boton.grid(row=fila, column=col, padx=5, pady=5)
```

7.1.2. Explicación del Código

1. Inicialización de la clase Calculadora

```
class Calculadora:  
    def __init__(self, ventana):  
        self.ventana = ventana  
        self.ventana.title("Calculadora")  
        self.i = 0 # Contador para la posición del texto  
  
        # Entrada de texto  
        self.e_texto = Entry(self.ventana, font=("Calibri 20"))  
        self.e_texto.grid(row=0, column=0, columnspan=4, padx=5, pady=5)  
  
        # Crear los botones de la calculadora  
        self.crear_botones()
```

- Se define el **constructor** `__init__()`, que se ejecuta cuando se crea una instancia de la clase. Aquí, se inicializa la ventana, se configura el **Entry** (el cuadro de texto donde se verá la operación), y se llama al método `crear_botones()` para generar todos los botones de la calculadora.



2. Definición de métodos

- **click_boton()**: Inserta el valor del botón presionado en el cuadro de texto.

```
def click_boton(self, valor):  
    self.e_texto.insert(self.i, valor)  
    self.i += 1
```

- **borrar()**: Borra todo el contenido del cuadro de texto.

```
def borrar(self):  
    self.e_texto.delete(0, END)  
    self.i = 0
```

- **hacer_operacion()**: Evalúa la expresión en el cuadro de texto usando la función `eval()`. Si hay un error en la operación, muestra "Error" en el cuadro de texto.

```
def hacer_operacion(self):  
    ecuacion = self.e_texto.get()  
    try:  
        resultado = eval(ecuacion)  
        self.e_texto.delete(0, END)  
        self.e_texto.insert(0, resultado)  
    except:  
        self.e_texto.delete(0, END)  
        self.e_texto.insert(0, "Error")  
    self.i = 0
```

- **crear_botones()**: Este método crea todos los botones de la calculadora y los coloca en la cuadrícula con la función `grid()`.

```
def crear_botones(self):  
    botones = [  
        ('1', 4, 0), ('2', 4, 1), ('3', 4, 2), ('4', 3, 0), ('5', 3, 1),  
        ('6', 3, 2),  
        ('7', 2, 0), ('8', 2, 1), ('9', 2, 2), ('0', 5, 0), ('.', 5, 1),  
        ('+', 3, 3), ('-', 4, 3), ('*', 2, 3), ('/', 1, 3),  
        ('(', 1, 1), (')', 1, 2), ('AC', 1, 0), ('=', 5, 2)  
    ]  
  
    for (texto, fila, col) in botones:
```

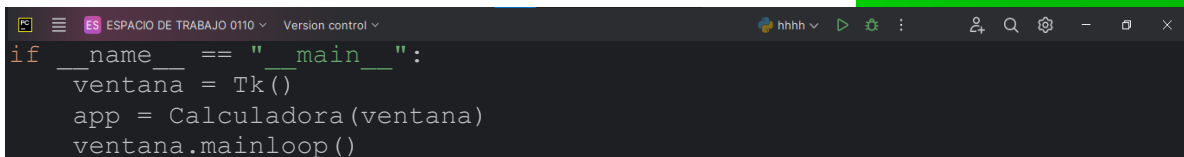



```
        if texto == 'AC':
            boton = Button(self.ventana, text=texto, width=5, height=2,
                           command=self.borrar)
        elif texto == '=':
            boton = Button(self.ventana, text=texto, width=5, height=2,
                           command=self.hacer_operacion)
        else:
            boton = Button(self.ventana, text=texto, width=5, height=2,
                           command=lambda t=texto: self.click_boton(t))

        boton.grid(row=fila, column=col, padx=5, pady=5)
```

3. Iniciar la calculadora

Para ejecutar la calculadora, simplemente creamos una instancia de la clase `Calculadora` y llamamos al método `mainloop()` de Tkinter:



```
if __name__ == "__main__":
    ventana = Tk()
    app = Calculadora(ventana)
    ventana.mainloop()
```

7.1.3. Proyecto: Calculadora con Tkinter

Objetivos del Proyecto:

- Implementar una **interfaz gráfica** con Tkinter.
- Usar la **Programación Orientada a Objetos** para organizar el código.
- Practicar el manejo de **eventos** con botones y el procesamiento de operaciones matemáticas.

Funcionalidades:

- Botones numéricos y de operaciones (+, -, *, /).
- Función para **borrar** el contenido.
- Evaluar expresiones matemáticas y mostrar el resultado.

Este proyecto permitirá a los estudiantes entender cómo integrar la lógica de Python en una aplicación visual.

