EXPLICACIÓN DEL PROGRAMA SOBRE VENTANAS CON POO

Importamos la librería estándar de para interfaces gráficas:

```
import tkinter as tk
```

Ahora podremos hacer referencia a los elementos típicos de la programación con interfaces gráficas: tk.Label, tk.Button,...

A continuación definimos una clase base para usar ventanas (partiendo de esta clase crearemos otras).

Creamos la ventana principal del programa con GUI , a continuación le damos un título y definimos su dimensiones.

```
class VentanaBase:
    def __init__(self, titulo):
        self.ventana = tk.Tk()
        self.ventana.title(titulo)
        self.ventana.geometry("300x200")
```

Dentro de la clase creamos un método que muestra la ventana.

```
def mostrar(self):
    self.ventana.mainloop()
```

A continuación creamos una clase hija de la anterior (hereda de ella):

```
class VentanaSaludo(VentanaBase):
    def __init__(self):
        super().__init__("Saludo personalizado")
```

Definimos el método constructor, que se ejecutará cada vez que creemos un objeto ventana. Con `super().__init__()` llamamos al constructor de la clase padre. Le pasamos como argumento el texto entrecomillado.

A continuación creamos una etiqueta de texto y hacemos que forme parte de la ventana principal e indicamos el texto de la etiqueta.

En la última línea hacemos que se muestre la etiqueta por pantalla usando un gestor de geometría (pack) y añadiendo un espacio vertical (padding) en el eje y de 10 píxels, así evitamos que se pegue a otros elementos.

```
#Etiqueta inicial
self.etiqueta = tk.Label(self.ventana, text="Introduce tu nombre:")
self.etiqueta.pack(pady=10)
```

Ahora creamos otro elemento: un cuadro de texto, que es un elemento básico de introducción de datos.

```
#Cuadro de texto
self.caja_nombre = tk.Entry(self.ventana)
self.caja_nombre.pack(pady=5)
```

De forma similar a los casos anteriores, ahora añadimos un botón para que al pulsarlo aparezca un saludo:

```
#Botón que genera el saludo
boton = tk.Button(self.ventana, text="Saludar", command=self.saludar)
boton.pack(pady=10)
```

En `command=self.saludar` le decimos a Tkinter que cuando el usuario haga click en el botón se ejecute el método saludar().

En las siguientes líneas se crea la etiqueta y se le da un valor vacío por defecto; después colocamos la etiqueta en la ventana.

```
# Etiqueta donde aparecerá el saludo
self.etiqueta_saludo = tk.Label(self.ventana, text="")
self.etiqueta_saludo.pack(pady=10)
```

Ahora vamos a crear un botón que llamará al método limpiar:

```
# Botón para limpiar
boton_limpiar = tk.Button(self.ventana, text="Limpiar", command=self.limpiar)
boton_limpiar.pack(pady=5)
```

Definimos un botón salir para cerrar la ventana:

```
# Botón para salir
boton_salir = tk.Button(self.ventana, text="Salir", command=self.ventana.destroy)
boton_salir.pack(pady=5)
```

A continuación creamos el primer método de la clase VentanaSaludo:

```
def saludar(self):
    nombre = self.caja_nombre.get()
    if nombre.strip() == "":
        self.etiqueta_saludo.config(text="Por favor, escriba un nombre")
    else:
        self.etiqueta_saludo.config(text=f"Hola, {nombre}!")
```

Se usa el método strip() (típico de strings o cadenas de caracteres) para quitar los posibles espacios en blanco en los extremos de la cadena; si después de eliminarlos la cadena es igual a nada, entonces es que la cadena estaba vacía, en ese caso pedimos al usuario que escriba un nombre, en caso contrario, asignamos a la etiqueta el texto de saludo con el nombre proporcionado por el usuario.

El segundo método de la clase VentanaSaludo sirve para borrar el contenido de la caja de texto eliminando los caracteres desde el inicio del texto, 0, hasta el final: tk.END.

Además se sustituye el texto de la etiqueta por una cadena vacía (otra forma de borrar).

```
def limpiar(self):
    self.caja_nombre.delete(0, tk.END)
    self.etiqueta_saludo.config(text="")
```

Por, último, y dentro del programa principal (NO dentro de la definición de la clase) creamos una instancia de la clase VentanaSaludo (el objeto se llama app) y hacemos que se muestre llamando al método mostrar().

```
app = VentanaSaludo()
app.mostrar()
```